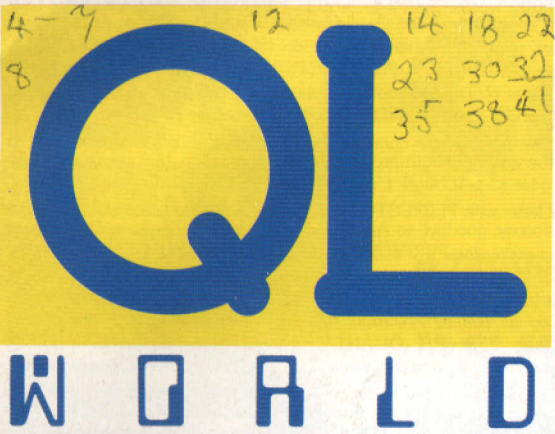


SINCLAIR



Volume 2 Issue 8 Price £2.50

QL Scene:
Air Message,
Plus Lots More

QL Bulletin Board

Trouble shooter:
QXL and SMSQ

SuperBasic in Action:
File Requester

Triqlhex:
Hair-tearing game



Editor
Helen Armstrong

Publisher
Mark Kasprovicz

Advertising Manager
Jim Peskett

Creative Director
John Stanley

Graphic Artist
Stevie Billington

Magazine Services
Linda Miller, Frances Maxwell,
Pauline Wakeling

Sinclair QL World,
Published by Arcwind Ltd.
The Blue Barn,
Tew Lane, Wootton,
Woodstock,
Oxon. OX7 1HA
Tel: 0993 811181
Fax: 0993 811481
ISSN 026806X

If you have any comments or difficulties please write to the editor and we will do our best to deal with your problem in the magazine, though we cannot guarantee individual replies. Back issues are available from the publisher price £2.50 UK, £2.99 Europe. Overseas rates on request.

Subscriptions from: Arcwind
The Blue Barn, Tew Lane,
Wootton, Woodstock, Oxon.
OX7 1HA
UK: £23.40
Europe: £32.90
Rest of World: £40.90

Reprographic Services:
Eclipse, Brook Street,
Watlington, Oxon. OX9 5JH.
Distributed by: Seymour Press
Ltd., Windsor House, 1270
London Road, Norbury,
London, SW16 4DH

© 1993 ARCWIND LTD.
Sinclair QL World is published 12
times a year. All rights reserved.
Reproduction in whole or in part
without written permission is
strictly prohibited. We welcome
contributions. All material must be
supplied with a SAE.

While all reasonable care is taken
in compiling QL World, the
publisher and its agents assume
no responsibility in affects arising
therefrom. Opinions expressed
are those of the authors.

Contents

VOL 2 ISSUE 8

4 TROUBLESHOOTER

Bryan Davies has some hard news on the QXL and SMSQ.

11 QL SCENE

More details on the QXL

12 OPEN CHANNEL

Compression and C68 ... QL and Organiser ... Help for Thor? ... Mice work ... Sidewinder ... Hermes ... EuroXchange changes ... Canada call.

16 INSTANT ACCESS

Fully updated guide to your software and hardware suppliers.

17 QL SCENE

More updates from Jochen Merz Software

18 DIY TOOLKIT

Simon Goodwin with a batch of short extensions this month.

22 QL SCENE

SJPD ... QReview ... Mersey Mouse ... Quanta show

23 THE NEW USER GUIDE - part 26

The Keyword Index from RJOB to SELECT ON

27 REVIEW: QUICKLASER

Bryan Davies with DP's printer driver for Professional Publisher.

30 QL BULLETIN BOARDS

Hilary Snaden on getting acquainted over the 'phone.

32 SUPERBASIC IN ACTION

Simon Goodwin embarks on a two-part file-requester.

35 HARDY HINTS

Getting data from one processor to another!

36 REVIEW: REMIND ME PLUS2

Henry Orlowski gets a new, smarter diary.

38 VERY BASIC SUPERBASIC

In Part 5, Dilwyn Jones starts on procedures and functions.

41 MACHINE CODE FOR BEGINNERS - PART 5

Alan Bridewell enters the murky world of hex, binary and logic.

45 CLUB ACCESS

46 MICRO ADS

Coming soon

Everything promised, and an article on Neural Networking and AI on the QL.

TROUBLE SHOOTER

Some concrete stuff on the QXL and SMSQ from Bryan Davies.

Version 5.08 of Perfection Plus Special Edition has been released. People who have bought Perfection SE versions 5.01 to 5.07 (with or without Spellchecker) can exchange their versions for 5.08, free of charge; they should send a stamped, addressed return envelope, or an International Reply Coupon to cover the cost of postage, and the original Perfection program disk, to Digital Precision. Make sure that any return envelope is strong enough for mailing a disk, and that the postage is sufficient. Digital Precision state that version 5.08 has fixed "all known bugs".

QXL News

This comes in two forms - "a little birdie said" and "from the horse's mouth". You will have to guess which is which. The latest software for the QXL (as of end-July) supports floppy and hard disk drives, parallel port, the KEYROW command, Mode8, and 800 x 600 pixels resolution. Quill, Easel and Archive seem to run fine on the QXL, at a decidedly brisk pace. Perfection runs at about 4.5 times the speed it does on a Gold Card QL. Many of its commands were already carried out too fast to follow on the GC, so hang on! It is reported that one program (name unknown) has been checked at 11 times its GC speed. Conqueror (standard and Special Edition) works with the QXL, the performance of the SE version being very pleasing. The enhanced screen resolution is good news, but it will be of use only with those programs that can make use of it, such as some Pointer Environment pro-

grams. As Text87 Plus-4 makes use of the enhanced resolution available on the Atari, it seems likely it will be able to offer an improved display with the QXL also.

Miracle Systems have never been enthusiastic about running the Gold Card at more than its rated speed (by means of The Poke). They will presumably frown upon the idea of running the QXL at well above its standard 20 MHz, but a report says an increase of 30% over the standard has been tried, with success. One thing to remember when over-running chips in this way is that heat generation increases with operating frequency, and life expectancy decreases with increase in heat. That is, you may get improved performance but you may also find the cpu chip getting unreliable, and even dying.

QLSpeed, the Archive-based test written by Eros Forenzi, provides a useful benchmark for the QXL. Eros reports a figure of 29 seconds for this test, as compared to the 123 seconds of the standard Gold Card. The only QXL I have seen working, so far, recorded 35 seconds. This is a fair performance increase, but it seems likely there is more to come. The Gold Card, when boosted by the POKE that increases effective clock speed to 24 MHz, records about 84 seconds. This reduces the QXL's advantage appreciably, but still leaves it well ahead.

What really matters to users is the way commercial software is handled. From my observations, Turbo Quill certainly runs very smoothly, the scrolling of text for example being both fast and steady. As one would expect, Perfection is very brisk, but there are some keypress problems to

be ironed out. Ambition, the business game, looks good, too. One aspect that is initially strange is the small space the QL screen occupies on a normal VGA monitor. The reason for the display not being fully utilised is that the transfer to the new display is pixel-by-pixel, with the result that only 512 x 256 of the possible 640 x 480 pixels on a VGA screen are used. Users of some Pointer Environment programs should already be benefitting from the greater resolution those programs support.

Switching between MS-DOS and the QXL is prompt and apparently reliable, and the QXL can be used under the Windows GUI. It is reported that the QXL also works under the DESQview interface, which is good news because that is a much more efficient program for multi-tasking and switching than Windows is. There are several things to be sorted-out in the running of SuperBasic, and of compiled programs; default directory access is said to be a bit uncertain, too. More importantly, there was no sign of trouble accessing both floppy and hard disk drives.

And more

A few more odd QXL reports, courtesy of Gerard Phelan: "Dave Walker is using 800 x 600 resolution successfully, with Pointer Environment programs. He is finding text hard to read on a 14-inch display at this resolution, but says it is fine on a 17-inch display. This is the same as I have found when using Windows, and it has forced me to get a 17-inch display; this has made life much easier for my

eyes, but at a high price.

"C68 3.05 programs run correctly (the report said that "all" such programs run correctly, but that is too brave a claim, surely?) There is no need to re-compile C68 programs."

The wide range of hardware to be found in PCs will inevitably create a few hiccups with the QXL, but Miracle are aware of some potential problems and are working on them. As is their normal practice, they intend to operate a generous software upgrade policy. It is clearly easier and, hopefully, cheaper to send out a disk with new software than to send out a rom chip, as had to be done with Trump Card and Gold Card. The hardware seems to be working satisfactorily, so early buyers should not be placed at a disadvantage. The main thing is that the QXL is now available.

Ram prices - a political question in the UK

Think carefully before deciding on how much memory to buy with your QXL. The price being charged for extra ram, above the basic 1 MB, is a fair one. Buyers saving a few pounds now, on the basis of installing extra ram chips later, may find it costing them a lot more than they planned on. Memory prices are subject to a number of major factors, obvious ones being the current level of world-wide demand, and import tariffs. The written word of the present time suggests demand for rams are high, and tariffs are either already here or on the way; that is, memory prices may go up in the near future.

Our "free market" EC has imposed a tariff of about 5 to 45% on imported floppy disks. Are there any non-imported floppies? A similar tariff is presumably still in force on printers, and it seems reasonable to assume memory is treated similarly. Wherever there is an "indigenous source of supply" to be propped up, the EEC imposes duties, which the consumers - us - pay. The computer buyer may never have heard of a European chip or disk manufacturer, and the printer names we know are not European either, but the manufacturers we may never buy from are being supported by tariffs (and subsidies, of course). Whatever your politics, bear in mind that ram prices have been low for the past few years, but have edged up noticeably this year (only partly because of the devaluation of the pound sterling).

SMS

There are no doubt other users who are somewhat unclear as to what the "SMS operating system" is. We have heard of it in connection with the Atari QL emulator, the QXL card, and a project (or projects) to develop a new super operating system, but there does not appear to have been a clear statement of what the various sets of letters signify. Unusually, someone has now taken the step of printing a single A5 information sheet, about SMS2, making a part of the story understandable. It is made clear that SMS2 and SMSQ are two significantly different things, the former being developed on the Atari as a more advanced OS than Qdos, whereas SMSQ was developed for the QXL, to be a Qdos clone. SMSQ very likely went via the Atari to get to the QXL, however. The SMS2 project name given is "Ora". The starting price for the OS on rom is quoted as £99. Remember that it is for the Atari only. One item notably absent from the document on SMS2 is any individual or company name, to identify its source and where to go to get the rom.

(Some of these points have appeared in past QL Scenes after conversations with Miracle Systems and others, but this is the first explanation we have seen quoting a price and project name.)

As is fairly normal for enthusiastic advocates of an enhanced Qdos, space is devoted to "knocking" existing (and highly successful) operating systems - dismissive mention is made of Unix, OS/2, (Windows) NT, Solaris and Mach. The fact that Mach is bracketed with the others suggests some lack of understanding of what it is (a "micro-kernel" which sits between the hardware and operating systems such as NeXTStep). On a more positive note, it is stated that hardware supported includes serial and parallel ports, floppy disk drives and SCSI hard disk drives, ramdisk, keyboard and mouse, MIDI (music) ports, and the Atari display formats 600 x 400 mono and 1000 x 900 4-colour. Facilities provided include advanced multi-tasking, high performance windowing, efficient inter-process communication, easy programming, and "response times better than the fastest real-time systems". That last claim takes some digesting!

Atari emphasis

A bulletin board message lists further project names - SMS3, SMS4, Karona, and Stella - linked to SMS2 and Ora. The slant is heavily towards Atari machines, partly because of the much greater numbers there are of them than of QLs. Whether or not the profusion of names, and the remarkable claims made, lead to anything concrete remains to be seen; the SMS2 as presently used on Atari variants clearly works, but it is not that far away from Qdos in its functionality, surely? Perhaps someone truly familiar with what is currently implemented would care to explain the features that are over-and-above those of Qdos?

In principle, the QL/Qdos user is being offered a path to much better things, rather than the

detail changes that have so far been available through Minerva. Maybe, after the false dawns of projects such as Futura and Thor 20/22 several years ago, we are about to get something real. This is certainly what many users would like. The question is, exactly what is available now? Meanwhile, the Miracle QXL is on the market.

Whichever way you look, it is towards a non-QL host - Atari or PC, with the Amiga a possibility. None of these routes can have a bright commercial future without attracting users from worlds other than the QL one. There are simply too few active QL users to support expensive software and hardware development on a large scale. In view of this, it is surprising that there does not appear to have been any advertising campaign to sell the QXL to PC users, for instance. As with everything in our micro world, it is the software that sells products. Somewhere along the line, users who presently know little or nothing of the QL will have to be told that QL software offers them something they cannot get (at reasonable cost) elsewhere.

Readers' letters

I M Gaye sent in another two disks for checking. This time, both were 3.5-inch DD, and both were bad. No amount of persuasion had any effect on either of them; for a change, it was definitely the disks themselves that were at fault. Gaye made the point that a friend had advised him against cleaning disk drives internally. This is a comment heard fairly often, the basis for it presumably being that disk cleaners can be abrasive and may cause wear to the read/write heads. There may be some justification in this, but I have not so far detected any problem resulting from my own use of disk drive cleaners, and my drives have been cleaned regularly (about once per month) for years. Some cleaners use dry paper, which has the appearance of filters for coffee machines, and this certainly worries me a bit, so I always put a

few drops of isopropyl alcohol on the cleaning disks before using them. This liquid is what is supplied with some commercial cleaning kits, and it can be obtained from chemists' shops; 50 ml is enough to last most users for a lifetime, and that volume cost me all of 46 pence.

Further problems with HD disks have led me - almost - to the conclusion that HD disks should never be given a DD format. Additionally, while DD disks in DD drives have rarely given me trouble, HD disks have given me trouble in both HD and ED drives, when formatted to both DD and HD. Nothing leads me to believe the problems are a function of the disks themselves, though.

Arvid Borretzen sent a hard disk back-up program from Norway, with a request that it be tested, as he does not have a QL with hard disk attached. At the moment, the hard disk is still banished from my QL, but it will be reconnected to test the program (not straight away!). Apologies for using "o" instead of the correct character in the surname. Arvid, but articles go through various filtering processes that will remove any "odd" characters, so there was no point putting the correct character in (an "o" with a "/" through it). (This is as much to do with the way the commercial systems talk to each other as it is to the way the QL talks to them. If I can catch them, they go straight back in again. Editor.) He is leader of a food and water testing laboratory, and says all the software used for the testing work is written in SuperBasic and compiled with Turbo or QLiberator.

The PC question

H F Banks posed the question of what he should spend his money on. He has in mind buying a QXL card, but knows little about PCs and wonders what sort to get to house the QXL. Naturally, he wants to continue using his current QL applications. One piece of advice given to him was to buy the minimum PC necessary to allow the QXL

to function, but he obviously feels that may not be the best thing to do. Initially, he thought that the 8 MB of ram available on the QXL could be shared with the PC, but he has discovered that this is not true; the memory on the QXL, and that in the host PC, are quite separate and neither accesses the other (it seems unlikely they will be able to in future, either). Unless absolute minimum cost is the prime consideration, it would be unwise to buy a "basic minimum" PC. Nothing is cheap for no reason, and a PC costing less than about £500 pounds can be expected to suffer some significant drawbacks, the main one being a hard disk drive unlikely to have greater than 40 MB capacity. Yes, there are - occasionally - bargains to be had from shops selling obsolete stock. Very rarely, you might get a better deal secondhand than new. For most QL users who want to experiment, and do not know the ins and outs of another market, the best thing is to buy new, and not get the bottom-of-the-range model either.

Banks also asked where he could get a cable made up, to connect the TTL/RGB socket of his Philips CM8833 monitor to the QL RGB socket. Try calling Tony Firshman at T F Services.

What an expert!

What is said in program reviews will always be the subject of some argument, and Stanley Hurwitz (I read elsewhere) would like to see articles by a "scrupulously fair-minded software expert", with (is he implying?) no commercial axe to grind. That is, someone who is in no way connected with the QL business. Well, most reviewers have no commercial connection with the QL business anyway. Occasionally we get a bit of free software, but when this happens it is usually either a beta-test version or an upgrade. And if we do get a new piece, then we have to learn it before we review it - there is no certainty that it is something that we will wish to use in our daily work! There is also an idea that we simply

accept what suppliers tell us. Well, we can't report everything that goes on behind the scenes!

To some extent, the wish makes sense, but who, outside of the QL world, would be suitably equipped with knowledge of any QL product to write a sensible review? Let alone a comparative one? You need to use the product for some time to do it justice. Equally importantly, you need to be aware of what the QL is, and has been through, and what else is available on it. If you are a keen reader of reviews in a variety of computer magazines, you will often spot writers who clearly do not normally use the product they are reviewing. In fact, the busy "hack" probably knows well very few of the products written about, in the sense that he or she can not - from sheer shortage of time - get to know more than, say, three or four major programs thoroughly.

Consider the two main word processing programs for the QL, Text87 and Perfection. They are quite unlike each other in many ways, despite doing basically the same job; the instructions for each are comprehensive, and take a long time to digest; both programs have gone through a long string of new versions; both are similar to Quill in certain respects (they are designed to be), leading to the possibility of getting mixed up about which program has which features, or handles a command in a particular way. Reviewers should try to be reasonably impartial, but they will naturally have preferences, and this is likely to show in what is written about programs. The reader has to read between the lines to get the full meaning.

Another sentiment voiced often to QL World and the Quanta newsletter is the desire for articles to be written for the person who is not "computer literate". This is easier said than done; for a technically-knowledgeable person to be able to put him or her self into the shoes of a complete beginner is quite a feat. You never quite make it, however hard you try. It is also slow work. A recent training session for a client who has

virtually no computer knowledge is still clear in my mind; the lady in question, and her husband, spent £1,000 on a system a year ago, and had then been given a single day of training by a "computer expert". After that, they left the system unused, simply because the information supplied with it had not enabled them to get to grips with it, and the trainer had talked "over their heads". Nothing of what she said remained in their minds. Although computer and software suppliers are often very much to blame for not writing their instructions for the type of person who has to read them, the users are as bad, for not making the effort to read what is written. New users want computing to be simple, but alas, it is not like that. You have to work at it.

A new approach

P H Tanner comes last - not because he wrote the most recent letter, rather the reverse, as his letter arrived in the middle of June. He does not write often, but what he does write takes a lot of digesting, and this latest letter has been re-read more than once. What prompted this missive was my query about who would buy the QXL card. He says he will, and maybe two of them, at that! Having said that, he wonders whether anyone else will buy one, since he considers himself to be such an unusual QL user that he cannot imagine anyone else wanting what he wants. A man who has said, more than once, that he really has no use at all for disk drives is certainly different from most computer users! His QLs are plugged together in a network, and spend their days (and nights) working on a joint project that requires relatively little access to microdrives. They are doing basic computing, akin to what "real" computers used to do, before the advent of micros, word processing etc.

The attraction of the QXL to Tanner stems partly from the fact that he did not feel the need for disk drives, and never bought a disk interface for that

reason. So, no disk drives, no Trump Card and no Gold Card. He feels he saved some money and can now spend it on the QXL. To make the decision easier, he has inherited an old 286 PC, which can be the host machine for a QXL. He envisions the QXL doing the "number crunching" work, with one QL driving a display, another QL being used for word processing, and all of them linked together through the PC. Carrying this integration theme a (large) step further, he thinks of putting two QXL cards into the one PC, sharing the common bus. Then, maybe, a 40 MHz version of the QXL. Not lacking in ambition, is Mr. Tanner! But he may not be that much different to quite a few other potential buyers of the QXL.

INFORMATION

TF Services has moved to:

Holly Corner
Priory Road
Chavey Down
Ascot
Berks SL5 8RL

Tel: 0344 890986.

TROUBLE SHOOTER

QXL - Air Message?

Rumours that Miracle's Stuart Honeyball and TF's Tony Firshman flew to America in June via "Hoover tickets" have been quashed by Stuart. "No such luck!", he says, adding that he hasn't been buying Hoovers recently, either.

Pity such a good story has no substance in it, but if someone of Stuart's capabilities could fly on a cleaning device, we doubt it would be broomstick, or even a Hoover - more likely a Vax!

Back on the ground, Stuart gave QL World the latest position on the **QXL**: "We tell our current customers what the QXL does and doesn't do at the moment," he says, "and as each new version of the software becomes available, the disk goes to QXL owners free of charge. So no-one loses out if they buy now rather than later."

"They'll get a few free floppy disks!" he adds cheerfully.

What about speculation that the QXL will run better on 486 machines than 286s?

"Our software is designed to run on the [generic] 8086," he says. "The size of the processor is not the issue. And it'll run on most BIOSs. There are a few un-solved things where we don't know if it's the BIOS or not, but we're going after those." PC users say that this is nothing new - moving between one "clone" and another often throws up problems caused by small but vital differences in the BIOS, even where the machine is described as "IBM-compatible".

In Memoriam - Ernie Wider, NESQLUG

Peter Hale

Ernie died on May 31, 1993, Memorial Day in the United States, the day we honor those who died in defense of our liberty.

He died of cancer after a long, hard struggle to beat the odds. He beat them - by all accounts by more than five months - because he had things to do.

In the end he was surrounded by his family at home. His daughter, Jennie, called me within the hour.

I write this because he was special to me, but I also write for the Sinclair computer press because the QL was special to him.

In his obituary, after the account of his career and his work with his church, it said that he was president of the New England Sinclair QL Group (NESQLUG).

Ernie rarely ventured outside the Psion suite that came with the QL; half the lines of programming he wrote were dictated over the phone, and he struggled to key in every line of SuperBasic from QL World. But he was the only person I know who had working skills with all four applications.

When he worked with the New England Telephone Company, he fought the trend and made the company get him a Thor. His boss justified it on the grounds that Ernie was more productive with a Thor than any two other employees with IBMs. Besides, Ernie told me, his files were too important to be subject to the viruses and security leaks that plagued the networks at the time.

He was a deeply spiritual man of a faith rarely found in the world today. From the day he learned he had cancer, and even when he learned his odds, he did not fear death. As a youth he had made the nine First Fridays, and knew that if he dies in a state of grace, he would go straight to Heaven. "And that's what life is all about anyway," said Ernie.

He set about putting his affairs in order and teaching his children to use the QL. In February he bought a Gold Card, to aid in that task.

By April he was bedridden, and had the QL moved to his bedside. Each stroke was painful, either from the cancer, the paralysis or the frustration of trying to think through a cloud of painkillers. In the end his son had learned enough to keypunch from dictation.

Ernie did not suffer fools gladly; indeed, he suffered them not at all, himself the least. He never complained that the cancer would cheat him.

Many his age become fixed in their ways and ideas, but he was always open to new ideas.

Several years ago Ernie told me about a financial planning program he was involved in. I bit my tongue, but suggested what I felt was a more appropriate avenue.

At first he resisted the notion and I let it drop. But eighteen months later he had the grace to tell me that not only had he followed up, but had spent some of the profits on an IBM so his wife could take work home from the office.

In consequence, I became privy to and part of his planning.

Ernie was not unlike many people whom I have met through a shared interest in the QL. He differed only in that he was my friend.

Oh, God! Ernest Wider, I miss you.

Peter also wrote: "The last time I wrote, you responded by publishing reference to Ernie, his election and his health. Some of the extra five months he got was due to the affirmation he received in QL World."

QL World is proud to be able to acknowledge people like Ernie Wider who were dedicated to the QL and to the QL community, and thank Peter for his part in it.

Some people say that a computer has no soul, but who's going to tell Ernie to leave his QL at the gates?

OpenChannel

Open Channel is where you have the opportunity to voice your opinions in Sinclair QL World. Whether you want to ask for help with a technical problem, provide somebody with an answer, or just sound off about something which bothers you, write to: Open Channel, QL World, The Blue Barn, Tew Lane, Wootton, Woodstock OX7 1HA.

Compression

In June's Troubleshooter, Richard Kettlewell is incorrect if he thinks that compression archivers (the correct term for "compression/decompression programs") have never made it into the QL world. The latest versions of all popular compression archivers have been ported to Qdos thanks to the high compatibility of the C68 compilation system, coming very close to ANSI standard.

Arc and Unzip were the first archivers available for the QL, already some years ago. With the appearance of C68, many people including myself started porting free programs from other platforms to the QL, and since C programmers are often found to be active in electronic communications, archivers were the first programs that brought C68 to the QL scene. By now, the latest versions of Zoo and Zip can be used on the QL. LHA is represented by LHQ. That means that any archive file with a .zoo, .zip, .lzh or (partly) .arc suffix can be created and melted. Archives created on a QL can be read on other systems without any problems and vice-versa. For the LH-suite of archivers (LHARC, LHA, etc.) the

operating system identification for Qdos is registered in Japan!

Many more programs have found their way into the Qdos scene by being ported with the superb C68 and there is an exciting future. Thanks to the continuing improvements of QL hardware (performance, memory, hard disks), even very large and complex programs, such as interpreters of all kinds, ray-tracers, programmable editors and even TeX are becoming real on the little black box. Admittedly, often the Gold Card or a similar powerful setup (such as the Atari STE with QVME) is required - but you are not investing in a Gold Card, Atari TT with QVME or even the extremely powerful QXL just to get the best of the Psion oldies, but also to run software that you couldn't run previously.

**Franz Herrmann
Ockenfels
Germany**

The C68 compiler is available from most Qdos public domain libraries.

Organiser

I was interested to see the letter from Lawrence Carpenter in 11.6 of QL World. I have been using both the QL and the Psion Organiser XP for business for many years. The only bottleneck in my system is the transfer of data collected during my travels on the XP to my QL on arriving home. I would appreciate any help you can give.

At home I use my QL with Gold Card, ED drives, Magnavox colour monitor, Epson LX-800 and Tandy DMP-137 printers and a Falkenburg keyboard interface with Apex keyboard. The working parts are

mounted in a nice mahogany case! I still prefer Quill for word processing and this is used on a daily basis. Archive is also used daily with the excellent Customer/Supplier database program from RMG Enterprises for statistics. My Organiser XP is used both at home and while travelling throughout Latin America and the Caribbean. It has the very complete LACE program from Beachcomber Software which allows me to handle business data, expenses with currency conversion, phone call timing, etc. Storage is on a 256K Flash Datapack. I use the old Transform Ltd. QL to Organiser program to back up data on the QL disk, but have not been able to get it to transfer data to Archive on the QL.

I look forward to seeing an article about data transfer between the QL and the Organiser in your excellent magazine.

**J Roy Goodall
Belize
Central America**

We will be making enquiries. I'm sure that if anyone else has been performing this useful transfer, they will be keen to share their method with Roy.

Thor help!

Please can anyone help me with the repair of a Thor XVI, which is not sending a proper signal to the monitor. Apart from this, all works well. I will travel a reasonable distance if necessary.

**Brian Richardson
Wallington
Surrey**

Most of the established Thor assistance is on the Continent.

Anyone nearby who can help, please write to Brian c/o QL World.

Blind mice!

I have recently been reading various articles in QL World and Quanta magazine about adding a mouse to a QL system, and I thought my experience in this "connection" would be of interest to some, so I am writing to both magazines.

At the Quanta AGM I bought a serial mouse from W N Richardson (EEC), since I was given the impression it was a straightforward plug-in device. It didn't take long to discover it was not that simple, as the "rat" would not work when connected to SER2 with my printer connected to SER1. It was obviously the common +12V power supply problem which I must say is not stressed very much in the literature supplied.

The professional way of overcoming this problem is to remove the main circuit board and fit an extra 680-ohm resistor in parallel with R10, but this involves making track cuts.

I used the following technique:

1. Disconnect all externals from the machine and remove the top case. Prop it up to remove stress on the ribbon cables and LED wires.

2. Drill a small hole through the back edge of the bottom case just to the right of the SER1 connector (as you look down on the machine).

3. Mount a small 2-way terminal block on the outside of the case, just to the right of the hole.

4. Locate C4 on the circuit board and solder a sort piece of red connecting wire to the positive axial wire of this component. Solder a short piece of black wire to a negative point on the

board. Pass these wires through the hole in the case and connect them to the terminal block. You now have a +12 volt supply external to the machine, capable of energising a small load.

5. Dismantle the D-connector of the short adaptor cable supplied with the mouse and disconnect the positive supply wire (coloured blue on mine) from pin 7. Connect a red wire to the positive terminal on the new block mounted on the QL and thread this wire to the D-connector. Fit a 680-ohm resistor between this wire and pin 7; there is just enough room. Tie back the original positive supply wire and reassemble the connector.

6. Reassemble the QL. This completes the hardware modifications.

7. Now load the software and amend any boot files as instructed and you should find your "rat" is now an all-singing, all-dancing mouse.

Having sorted out this problem in my own way I felt I should warn others of the difficulties involved in using the mouse in SER2. I feel that it is unfair to give misleading ideas about the ease of fitting the Serial Mouse. It is not a straightforward plug-in operation. When I phoned Richardson's to ask for advice they didn't seem to know much about it, and referred me to a third party who confirmed the real problem. So, prospective purchasers be warned, a Serial Mouse will not work on a standard QL with another device connected to SER1 until the machine is modified to provide a separate current-limited +12 volt supply.

Phil Stickley
Weybridge
Surrey

In the June (11.6) QL World, Bryan Davies tested and assessed the effect of the Serial Mouse on the QL's supply, and concluded that operation with a direct serial connection should not overload the supply, but operation with a serial-parallel interface (common in QL systems) would do so, and "appears

to rule out this form of mouse interface ... for many users" unless the recommended 680-ohm resistor is fitted. Bryan also goes into some detail about configuring cables to allow a functional connection between the mouse and a standard QL for DIY purchasers.

W N Richardson (like other hardware retailers) sell a wide range of hardware and they can't be expert on everything!! They do what any responsible dealer does - if they don't know the answer, they refer you to a man who does!

Opinions as to whether Phil's mod is less drastic than adding an internal resistor and cutting tracks will vary - each to his own! Beware of undertaking hardware mods unless you know what you are doing; you do them at your own risk, and may invalidate a warranty or damage the machine. W N Richardson, or someone else, may know a man who can do a mod for you.

Sidewinder

For those who are interested in sideways printing, Dilwyn Jones Computing has recently announced a new program, Sidewinder, which might fit the bill. It can print wide spreadsheets or text in landscape mode on 9-pin and 24-pin printers.

Alternatively, you could consider upgrading to an HP Deskjet printer, which supports landscape printing in three character widths. The printer rotates the characters itself, so there is no need to send a slow graphics dump, and landscape printing is almost as fast as conventional upright "portrait" text output; there's a slight overhead as it scans the entire page rather than the columns where ink is needed, but it's much faster than a graphics dump, not least because the amount of data sent from the QL is much reduced.

You do not need to fiddle with INSTALL or PRINTER.DAT to

sue HP landscape mode. Turn on the printer and press the FONT button four or five times to select 183 or 220 columns respectively. Alter the P and L options in the Abacus DESIGN menu to tell the program the new page size, then PRINT as normal.

Simon N Goodwin
West Midlands

Modems and Hermes

Thanks for the TF Services mentions in recent issues. The reason for the computer in the loo was that I needed earthed US sockets and there were none in the bedroom! I must say, though, the photo was engineered by me.

It was a welcome review of Hermes by someone who uses

and appreciates it. It's only a pity Rich missed the whole reason Hermes was developed in the first place. Hermes can support 19200 bps input and thus run any modem - at least, we and our customers have not yet found one that does not work. All the other features which he praises were added during development, rather than being part of the brief to Laurence.

On another subject, please spread the word about our new address, which is Holly Corner, Priory Road, Chavey Down, Ascot, Berks SL5 8RL Tel. 0344 890986.

Tony Firshman
T F Services
Ascot

I've heard several explanations as to why the computer was in the loo, and several people claiming to set up the photo! But Tone is clearly the power before the Throne.

Editor's notebook

Never fear, Mike Lloyd's back. The New User Guide is on page 23-26 as "usual". In fact, next month is looking good, as his next column's just arrived a little in advance!

The postbag always goes quiet in August as Great Britain's QLers pack their bags and head for the sea (or more likely their back gardens, the way things are with many people - but the principle's the same. No more crouching over keyboards for a week or two, by order). But we have had some welcome letters from readers overseas. Hugh H Howie in particular (his letter arrived this morning) is talking about the vast distances travelled and the "Texan-sized" fairs in the North Americas. No wonder some of the UK traders are looking that way. QLers may be thin on the ground compared to some, but it's a big place.

OpenChannel

EuroXchange

After reading your news about Xchange (QL Scene, QL World II.6) I requested it from Gunther Strube, who sent the disk to me immediately.

I found that Quill and Abacus don't read any kind of files (.doc or .aba) but Archive and Easel seem to work well. I have written to Gunther on this problem.

Then I examined the three utilities, PRinto, Pedit and Replace: Pedit and Replace work well, they can be compiled by Turbo with no problems. I only remarked the line, that has statement SYS_VARS=163840, because SYS_VARS is a Turbo function.

PRinfo doesn't work, apparently because it has some errors during the handling of the serial printer.

The program halts on fields PARITY and PAPER FORMAT, and I must change the lines 2220 FOR loop=1 to 5: READ paritet\$(loop) and 2400 en_til 1,1:PRINT#7,betegnelse\$(38);To 20 (etc.) to read:

```
2220 FOR loop=0 TO 4:
READ paritet$(loop)
2400 en_til 1,1:PRINT#7,betegnelse$(28);To 20 (etc.)
```

because the index of array paritet\$ starts from 0-NONE and finishes to 4-EVEN (1-LOW, 2-HIGH, 3-ODD) and the max value of betegnelse\$ is 28 and not 38.

If the printer is a parallel, it works well.

You must add on line 1140:CLOSE#5 for closing Xchange_dat, which remained open otherwise.

If you have installed Turbo Toolkit, you must remark line 1040 SYS_VARS=163840. You could remark or delete the line 1020 DIM prog_use(40), dat_use\$(40), dev_data\$(40), dev_prog\$(40), because it's a duplicate of line 1000. With the inserting of a few lines and the changing of others, the program can be improved: the filename to examine and the drive from read are requested from console.

The changed lines are:

```
1000 DIM P_tekst$(222), (etc.)
1      2      2      0
OPEN#9,con_300x18a116x202
: (etc.)
1900                                IF
bested=0:PRINT#kanal,!(... Esc -
Quit, Continue - ENTER,
Change=F1 .)ELSE (etc.)
1940 IF tryk=10 or tryk=27 or
tryk=232: EXIT loop
```

The inserted lines are:

```
1      0      4      5
xchange$="xchange_dat";
REMark this is the default
1425 IF esc_flag=232 :
change_device
1445 IF esc_flag=232 :
change_device
1      7      3      5
p_tekst$=p_tekst$&chr$(10)&"Y
ou can replace device and text
printer name"
2      1      2      5
xchange$="xchange_dat";
REMark restore default and the
new procedure
CHANGE_DEVICE
4000 DEFine Procedure
change_device
4001 LOCAL f$,an$
4002 CLS#9
4003 INPUT#9,"Device : "f$;
REMark ENTER default
4004 IF f$="-" then GO TO
4007
4005 IF f$-<>"4" then goto
4002: REMark only FLP1 or
FLP2
4006 dev_data$=f$&"_"
4007 CLS#9
4008 INPUT#9,"Driver printer :
"an$
4009 IF an$="-" then GO TO
4012
4010 IF LEN(an$) > 36 then
GO TO 4007
4011 xchange$=an$
4012 vis_pdriver
4013 PRINT#7,p_tekst$
4014 vent 9,0
4015 END DEFine
change_device
```

Gian Paolo Marcolongo
Cesena
Italy

This version of Xchange is not identical with the one that Simon Goodwin described in QL World II.7 last month, which is becoming available from UK public

domain libraries.

Thank you to Gian, who also takes the biscuit for the faintes printer ribbon of the year so far!

Long distance

I enclose a report on my visit to the Newport fest in June, mention the feasibility of a fest here next year. This is off the top of my head, and I do not know if I can get the club (the Toronto Timex-Sinclair Users Club of Canada) to go along with me. The first step is to get some feedback.

I will be attending an All Formats Computer Show in Dayton, Ohio at the end of August. A number of Sinclair clubs will be in attendance, and from that I should be able to get a feeling as to how many might attend a fest here.

Incidentally, this show in Dayton is the largest in the Mid West USA, and had an attendance of 27,500 last year. I covers just about every computer in use, with flea markets and traders all over the place, over 100 traders, the lot! Oh that we had a Sinclair fraternity like that. But at least "Sinclair" will be there.

My travelling distance to Newport was about 600 miles and my distance to Dayton is about 450 miles, so we do travel long distances where our hobby is concerned.

Hugh H Howie
Burlington
Ontario
Canada

Thank you for the subscription Hugh. This letter arrived just as we were going to press, so look out for Hugh's report next month.

Merz updates and bug-fixes

Jochen Merz Software are forging ahead with their regular software updates. The **QVME** QL Emulator can now even run "badly written or old software which writes directly to the QL screen, which is usually located at address 131072." Old games and graphics programs can be run on the emulator in this way: all programs open the windows in an "old" screen area, where "direct dirty accesses" are done. The lot is then copied a number of times per second to the "real" screen memory on the QVME card automatically by the Atari Blitter custom chip, and interleaved with the processor, so that copying does not slow the machine down much.

All advanced programs used the screen driver, so in most cases this "compatibility mode" will not be needed, but it's a help for fans of older software. Updates are free, and a new V6.4 manual is recommended. The QVME now runs on the 68030 Atari TT, with a fast floating point chip 68882, both running at 32 MHz. The TT version has the same features as the Mega STE versions, supports the same devices, including hard disks, and costs the same.

On the software front, **Menu Extension** has a new menu called Character Select, which allows character generation via a table. QD's integrated **Help** system has been improved again, with short-cut key-strokes and a DO to toggle window sizes. V5.00 owners get a free upgrade.

MenuConfig V1.18 has now fixed some bugs that crept in with upgrades in V1.14. Brainsmasher owners can get a free upgrade by sending their master disks back to Jochen. The popular QSpread is up to version 1.17 with many new features in this upgrade - number-to-text conversion (for printing on cheques, etc.), Find, extended grid modification commands including adjustment of cell references when rows and columns are added or removed. Merz reckon QSpread has been prettywell bug-free since version 1.15, and recommends owners of 1.12 or earlier to get a free software update and order a new manual from the company while they are at it.

Jochen Merz Software is, as ever, at **Im Stillen Winkel 12, D-47169 Duisberg, Germany**. Take a note of their German postcode for your address books.

Quanta trade charges

Quanta has decided to charge traders at its Workshops £30 on the door, or £20 if booked in advance, for trade stands. Developers who wish to exhibit but not sell products will be free of charge. A registration form with full requirements, including power points, is available from **Phil Jones, 66 Devonshire Avenue, Long Eaton, Nottingham NG10 2EP**.

The next Workshop is at Bristol Walton Park Hotel, Wellington Terrace, Clevedon, Avon on 17th October 1993.

The Bristol workshops have a popular reputation with traders and the public.

ALL FORMATS DIARY

Coming dates for the **All Formats Computer Fair** are:

August: no dates listed; Sep West Midlands National Motorcycle Museum, junction 6 of the M42; 12 Sep Brighton Corn Exchange, Church St, Brighton 18 Sep Leicester De Montfort Hall, Granville St, Leicester 19 North East Washington Leisure Centre, Dist 1 25 Sep Edinburgh Adam House, Chambers St 26 Sep Glasgow Mitchell Theatre, Granville Road. (Next Novotel, Hammersmith Fair is on October 23rd)

Check with suppliers whether they will be at a particular Fair. If you have far to travel phone All Formats 0608 663820 to check arrangements haven't changed.

Day tickets are £4; attendees can get up to 50 £1-off vouchers by sending an SAE to the organisers at Maple Leaf, Stretton-on-Fosse, Moreton-in March, Gloucestershire GL56 9QX. (Only one voucher per ticket.) Photocopies of these vouchers are also accepted. Admission to the Fairs is a flat £2 between 2pm and 4pm (£1-off vouchers do not apply at these times).

Silver Service

With software service companies in the PC-compatible sector now charging hundreds or even thousands of pounds for software support after a limited initial period of ownership (This is Help-desk support we're talking about. Upgrades are a different story again), Freddie of Digital Precision has been heard to mutter that DP provides software support free and for an indefinite period for its customers.

In fact, you can rely on the vast majority of QL suppliers now in the business to help out with any problems you may have while using their products wherever they can. There are some real benefits in being with the QL.

QL
S
E
R
V
I
C
E

DIY TOOLKIT

Three versatile file-handling SuperBasic extensions join Simon Goodwin's DIY Toolkit.

THE latest arrivals are file-handling extensions: the INPUT\$, ANYOPEN% functions, and the SET_POS command.

These extensions make direct calls to the QL operating system, permitting access to any file, new or old, including the Qdos directory hidden on every file-structured device. They let you move to any point in a file without traversing the intervening bytes, and read up to 32K at one gulp.

These are very useful, but simpler than recent DIY Toolkit projects, and may serve as an introduction to budding assembler programmers.

INPUT\$

INPUT\$ is a function to read a sequence of bytes from a file in one step. It takes two parameters - the number of an open SuperBasic channel, and a positive integer length. The function is present in **Microsoft Basic** and **Turbo Toolkit**, but it does not appear in QL roms, which is a shame as it is many times faster than the official alternative, repeated calls to INKEY\$.

INPUT\$ can swallow up to 32767 bytes in one mouthful. I use it on the Speculator file converters, which convert 48K snapshots of Spectrum memory from PC and Amiga to QL format. Snapshot files consist of a 48K memory image, plus extra information about the state of the processor and emulator. This extra information may appear at the start or end of the file, and its format varies between emulators.

INPUT\$ is an efficient way to move the main body of the file from one format to another. Even reading a modest 1024 bytes at a time, INPUT\$ is about 20 times faster than INKEY\$. This makes a real dif-

ference, even on a Gold Card, where snapshot conversion takes under ten seconds with INPUT\$ and more than three minutes with INKEY\$.

INPUT\$ is also a good way to read the 64 byte entry for each file in a Qdos directory. The Toolkit 2 string GET operation is not suitable in such cases, as it relies on a positive length word at the start of each string, and this will not appear in real binary data.

INPUT\$ is the most general way to read data into a string; it can deputise for string GET as long as you use INPUT\$(#chan,2) or GET #chan,len% to find the length before you read the string text.

SET_POS

SET_POS is a simple command to set the file pointer. As a file is read or written Qdos keeps track of the next byte to be transferred with a 'file pointer' - a long integer value between zero and the number of bytes in the file. When it is zero it is at the start of the file, so SET_POS #3,0 rewinds the file pointer to the start. This is useful if you have just written a file with OPEN_NEW or OPEN_OVER and wish to read it from the start without re-opening it.

Likewise, you can extend an existing file by opening it in the usual way with OPEN and moving the pointer to the end before you add new data. If you attempt to SET_POS past the end of the file, Qdos rounds the position down to the last valid position, so SET_POS #3,1E6 will reliably find the end of a file up to a million bytes long. The QL microdrive handler has a 24 bit limit on pointer position, so attempts to SET_POS beyond about 16,777,000 will cause a 'bad medium' error.

* QL WORLD DIY TOOLKIT - GENERAL-PURPOSE CHANNEL EXTENSIONS
* Version 0.7, Copyright 1993 Simon N Goodwin.

```
*
start      lea     define,a1
            move.w $110.w,a2      BP.INIT vector
            jmp     (a2)

* result = ANYOPEN% ( name$, mode$ )
*
anyopen    lea     16(a3),a4
            cmpa.l a4,a5
            bne     bad_param     Two parameters please
            addq.l #8,a3           Get integer mode first
            movea.w $112.w,a2      Vector to get an integer
            jsr     (a2)
            bne.s   bad_exit
            move.l  a1,d7          Save RI pointer for later
            move.w  0(a1,a6.1),d5  Save open mode for later
            subq.l  #8,a3
            subq.l  #8,a5          Recover first parameter
            movea.w $116.w,a2      Vector to get a string
            jsr     (a2)
            bne.s   bad_exit

*
* Attempt to open a channel using the string on the RI stack
*
            move.w  d5,d3          Restore Open mode
            ext.l   d3
            movea.l a1,a0          Take nothing for granted
            moveq   #-1,d1         Point A0 at the string
            moveq   #1,d0          Owner is this task
            trap    #4             IO.OPEN trap key
            trap    #2            A0 is A6-relative
            tst.l   d0             Try it!
            bne.s   return_d0

*
* It worked, find space for it in BASIC's channel table
*
            movea.l $30(a6),a2     Table entry size
            moveq   #40,d1         Past the end?
scan_chans cmpa.l $34(a6),a2
            bge.s   make_room
            tst.b   0(a2,a6.1)     Is it free?
            bmi.s   found_room
            adda.l  d1,a2          Try the next slot
            addq.w  #1,d0          D0 is channel number
            bra.s   scan_chans

*
make_room  move.w  d0,d6           Save BASIC # for later
            movea.w $11A.w,a2      Fetch base vector
            lea     44(a2),a2      Adjust for channels
            jsr     (a2)
            movea.l $34(a6),a2     Point at BV.CHP
            lea     40(a2),a3      Move up one entry
            move.l  a3,$34(a6)     Claim the new one
            move.w  d6,d0          Restore BASIC #

*
* Store channel ID and initialise BASIC channel details
*
found_room move.l  a0,0(a2,a6.1)
            moveq   #7,d1          Count for 32 bytes
```

You can move the pointer on most systems (apart from some Thor XVI and Minerva roms) with undocumented calls to PAN and SCROLL, documented in QL World February 1989, but this is obscure and complicated for long files, as those commands limit moves to 32K at a time. SET_POS is included here as it is an interesting example and a useful complement to INPUT\$.

Both routines

SET_POS and INPUT\$ are ideal for use when reading sectors from a disk. Use SET_POS and PRINT; to write

sectors back. They can also be used to get around a bug in early QL roms, which limit INPUT lines to 128 characters. AH and JM roms report a 'buffer full' error if you try to read longer lines.

If you use INPUT\$ to read groups of characters the 128 byte limit is ignored. The approach is to read groups of bytes, building up a long string, until the CHR\$(10) marker at the end of the line is found with INSTR.

At this stage you discard the rest of the string and use SET_POS to wind back to the character after the CHR\$(10), ready to read the next line.

Your program needs to keep track of the number of bytes read so far, but this is little trouble, as it corresponds to the number of lines read and their total length.

ate a new channel table entry. The method was introduced in my February 1992 QL World column, and is reliable on all known QL roms and emulators. The method can be used in

```
clear_table addq.l #4,a2
clr.l 0(a2,a6.l) Initialise turtle etc.
dbra d1,clear_table
move.b #80,3(a2,a6.l) Fix default WIDTH

*
return_d0 move.l d7,$58(a6) Update BV.RIP
move.w d0,0(a6,d7.l) Stack the result
moveq #3,d4 Type = 16 bit Integer
moveq #0,d0 No run-time error
rts

*
bad_param moveq #-15,d0 BAD PARAMETER error
bad_exit rts

*
* result$ = INPUT$( # channel$, bytes$ )
*
* Read and check both parameters: channel number & length
*
inbytes movea.w $112.w,a2 Vector to get integers
jsr (a2)
bne.s bad_exit
subq.w #2,d3
bne.s bad_param
move.w 0(a1,a6.l),d0 Channel number
bsr.s get_qdos_id Convert BASIC # to ID
move.w 2(a1,a6.l),d5 Bytes to be read
ble.s bad_param
addq.l #4,$58(a6) Unstack two integers

*
* Make room to read the string onto the RI stack
*
ext.l d5
move.l d5,d1 D1 is space needed
addq.l #2,d1 Allow length word
move.w $11A.w,a2 Find BV.CHRIX
jsr (a2) Allocate RI space
movea.l $58(a6),a1 Get BV.RIP
suba.l d5,a1 Move it down
btst #0,d5 Is the length odd?
beq.s its_even
subq.l #1,a1 Ensure a word boundary

*
* Fetch the string from the file to the RI stack
*
its_even moveq #-1,d3 Indefinite timeout
move.l d5,d2 D2 is length in bytes
trap #4 A1 is relative to A6
moveq #3,d0 IO.FSTRG key
trap #3
tst.l d0 Did it work?
bne.s bad_exit
suba.l d5,a1 Wind back over the text
subq.l #2,a1 Allow for a prefix word
move.w d1,0(a1,a6.l) Set string length
move.l a1,$58(a6) Set BV.RIP
moveq #1,d4 Indicate string result
rts
```

ANYOPEN%

ANYOPEN% is complementary to QTRAP, BTRAP and MTRAP, from DIY Toolkit Volume T; it lets you pass any string to the Qdos OPEN trap, TRAP #2 with D0 set to one. The function either returns the negative Qdos error code, or a positive number to indicate the new channel available to SuperBasic.

Many DIY Toolkit extensions include code to look through the SuperBasic channel table, but this is the first that can cre-

ate any extension that needs to open a file and make it available to Basic. I didn't know how to do this when I wrote the Turbo Toolkit CONNECT command, which is why the output pipe channel number is required to be lower than the input one!

ANYOPEN% takes two parameters - a string and an open-type, as documented on page 40 of Andrew Pennell's **Qdos Companion**. Type zero is used to open an existing file or device for reading and writing, like the normal OPEN. Type 1 corresponds to OPEN_IN, creating a read-only channel. Type

2 is OPEN_NEW, while types 3 and 4 match the Toolkit 2 commands OPEN_OVER and OPEN_DIR.

Original Sinclair JM and AH roms do not support type 3 on microdrive, but type 3 is fully implemented by Toolkit 2 as well as Minerva, JS and MG roms. If your code must run on all QLs you can simulate a type 3 OPEN with a simple sequence. If ANYOPEN%(FILES\$,1) works, CLOSE the file, DELETE it and try again with ANYOPEN%(FILES\$,2).

Error messages

If ANYOPEN% fails it returns a negative Qdos error code to indicate what went wrong. The values are listed in most QL books. Pages 19 and 20 of the QL User Guide (December 1984 update) give the codes and their meanings, although the list confusingly uses positive rather than negative numbers.

At first sight ANYOPEN% might look a dead ringer for QJump's FOP_ functions, but I have found it more useful for

checking devices because it does not try to prepend a default prefix if the name is initially rejected.

I have found it almost impossible to detect a missing drive with FOP_DIR on a Gold Card, because any reference to, say, MDV1_ is converted to FLP1_MDV1_ if your default is FLP1_, and FOP_DIR happily opens the floppy directory without indicating the absence of MDV1_. The default prefixes in Toolkit 2 are often convenient, but not in this case. If you really need to know if a directory exists, ANYOPEN% is less sophisticated, and more useful.

ANYOPEN% with parameter 2 is similar to the original DEVICE_STATUS%, part of the Supercharge toolkit. The Turbo Toolkit function of the same name is much more powerful, but it will not allow access to directories, and can cause delays and 'bad or changed medium' messages if you use it to check the status of a name on a write-protected microdrive.

The only way to be sure that a medium can be written is to

```
*
* SET_POS # channel , position
*
* Moves to start or EOF if 32 bit position is beyond file end
*
setpos movea.w $118.w,a2 Vector gets addresses
jsr (a2)
bne.s bad_exit
subq.w #2,d3 Test for two parameters
bne.s bad_param Reject otherwise
move.l 0(a1,a6.l),d0
bsr.s get_qdos_id

*
* Set FS.POSAB registers and exit via QDOS TRAP handler
*
moveq #-1,d3 Wait as long as it takes
move.l 4(a1,a6.l),d1
moveq #3,d0 Set the trap key
trap #3 Call FS.POSAB
rts Return D0 error, if any

*
* Convert channel number in D0 to channel ID in A0
*
get_qdos_id mulu #40,d0 Scale for table size
bmi.s what_chan Channel numbers start at 0
add.l $30(a6),d0 Add table base offset
cmp.l $34(a6),d0
bge.s what_chan Past end of table?
move.l 0(a6,d0.l),d0 Fetch ID from table
bmi.s what_chan Channel closed if negative
movea.l d0,a0
rts

*
what_chan addq.l #4,a7 Return to prior caller
moveq #-6,d0 CHANNEL NOT OPEN error
rts

*
define dc.w 1 One procedure
dc.w setpos-* # channel$, position
dc.b 7,'SET_POS'
dc.w 0
dc.w 2 Two functions
dc.w inbytes-* ( # channel$, bytes$ )
dc.b 6,'INPUT$'
dc.w anyopen-* ( name$, mode$ )
dc.b 8,'ANYOPEN%'
dc.w 0

*
end
```

try to write to it; there is no error when you open the file, even if it is 'in use' or 'read-only'. `DEVICE_STATUS%` with the 'try everything' -1 option attempts to re-write the first byte of an existing file. This works fine with disks, but the microdrive handler does not detect the missing tag. It repeatedly tries to write the sector, but the drive observes the write-protection and stubbornly refuses to play along, leading to the dreaded report a minute or so later. Don't panic - your data is intact.

`ANYOPEN%` does not try to write when it opens an existing file, so the error comes only when the user tries to re-write the file. This is not the ideal solution, but it's the best we can do without fixing the hardware.

SuperBasic checks

You can avoid most problems if you perform extra SuperBasic checks when you know you are not using a microdrive - a STAT report over 250 sectors is a reliable indication, and could be used in conjunction with a device prefix check and - Qdos willing - tests on system variables like `SV.MDRUN`.

Loading and GoingTo use these extensions you must either type in one of the listings, or obtain the equivalent on tape or disk by post from DIY Toolkit. Listing one is assembler source, tested with QL Devpac 2.0, while Listing two is a SuperBasic loader for corresponding hex data.

When you type in and run Listing two it checks the data and creates a small code file on any device you choose. Once you have created `FLP1.ACCESS.CODE` you can link the function to SuperBasic with these commands:

```
X=RESPR(342)LBYTES
FLP1.ACCESS.CODE,XCALL X
```

Source and binary code for these extensions has been added to DIY Toolkit Volume E, where they join `PURGE`, the

string select function `PICKS`, and the error trappers `CHECK%` and `CHECKF`.

24 volumes of DIY Toolkit goodies are now available on disk and microdrive, priced at £3 each on 3.5 or 5.25 inch Qdos disks, or £4 per volume on cartridge. Full instructions and technical notes come on the drive, in Quill .DOC format; order two or more volumes and you get laser-printed documentation for each volume at no extra charge. To obtain volumes, or further information, write to the DIY Toolkit librarian Dr. Bill Fuggle at DIY Toolkit, 86 Lordwood Road, Harborne, Birmingham B17 9BY, UK.

For an example of `INPUT$` and `ANYOPEN%`, see this month's SuperBasic in Action. I shall continue by discussing Listing one, which should already be quite clear from the comments alongside the code. The code is re-entrant and 'romable'. It has been tested on Gold Card JM and Amiga JS Qdos systems.

How it works

The `START` routine is only used when you first `CALL` the code. It links the new names at the end of the file to SuperBasic.

The assembly code for `ANY-OPEN%` comes next. It reads the second parameter (the open-type) first, to find out the value of `A1` that corresponds to a single integer on the stack, and copies this to `D7` as it will come in handy later.

The next call fetches a string onto the maths stack, and passes it to `TRAP #2`, trying to open it. If `D0` is not zero after this, the error code is returned to Basic in the space previously reserved for the open-type. The string is discarded as `BV.RIP` points past it when the function returns.

If the `OPEN` succeeds the program looks for a space in the SuperBasic channel table. Empty slots have a negative value in the first byte, and can be re-used.

If all slots are in use execution continues at `MAKE_ROOM`. Until last year

```
100 REMark Sinclair QL World HEX LOADER v 3
110 REMark by Marcus Jeffery & Simon N Goodwin
120 :
130 CLS: RESTORE : READ space: start=RESPR(space)
140 PRINT "Loading Hex..." : HEX_LOAD start
150 INPUT "Save to file...":f$
160 SBYTES f$,start,byte : STOP
170 :
180 DEFine Function DECIMAL(x)
190 RETURN CODE(h$(x))-48-7*(h$(x)>"9")
200 END DEFine DECIMAL
210 :
220 DEFine PROCEDURE HEX_LOAD(start)
230 byte = 0 : checksum = 0
240 REPEAT load_hex_digits
250 READ h$
260 IF h$="" : EXIT load_hex_digits
270 IF LEN(h$) MOD 2
280 PRINT"Odd number of hex digits in: ";h$
290 STOP
300 END IF
310 FOR b = 1 TO LEN(h$) STEP 2
320 hb = DECIMAL(h$) : lb = DECIMAL(h$(b+1))
330 IF hb<0 OR hb>15 OR lb<0 OR lb>15
340 PRINT"illegal hex digit in: ";h$ : STOP
350 END IF
360 POKE start+byte,16*hb+lb
370 checksum = checksum + 16*hb + lb
380 byte = byte + 1
390 NEXT b
400 END REPEAT load_hex_digits
410 READ check
420 IF check <> checksum
430 PRINT "Checksum incorrect. Recheck data.":STOP
440 END IF
450 PRINT "Checksum correct, data entered at: ";start
460 END DEFine HEX_LOAD
470 :
480 REMark Space requirements for the machine code
490 DATA 342
500 :
510 REMark Machine code data
520 DATA "43FA012C34780110","4ED249EB0010BBCC"
530 DATA "66000086508B3478","01124E92667C2E09"
540 DATA "3A31E800518B518D","347801164E92666A"
550 DATA "360548C3204927FF","70014E444E424A80"
560 DATA "6648246E00307228","B5EE00346C0C4A32"
570 DATA "E8006E20D5C15240","60EE3C003478011A"
580 DATA "45EA002C4E92246E","003447EA00282D4B"
590 DATA "003430C62588E800","7207588A42B2E800"
600 DATA "51C9FF8F815BC0050","E8032D4700583D80"
610 DATA "7800780370004E75","70F14E7534780112"
620 DATA "4E9266F6554366F0","3031E80061603A31"
630 DATA "E8026FE458AE0058","48C5220554813478"
640 DATA "011A4E92226E0058","93C5080500006702"
650 DATA "538976FF24054E44","70034E434A8066BA"
660 DATA "93C555893381E800","2D49005878014E75"
670 DATA "347801184E9266A2","5543669C2031E800"
680 DATA "610C76FF2231E804","70424E434E75C0FC"
690 DATA "00286B14D0AE0030","BCAE00346C0A2036"
700 DATA "08006B0420404E75","588F70FA4E750001"
710 DATA "FFC0075345545F50","4F5300000002FF5E"
720 DATA "06494E5055542400","FEC20841E594F50"
730 DATA "454E25000000","*",29540
```

there was no documented way to expand the SuperBasic channel table. This is the first demonstration of the required technique in assembly code.

It fetches the value of the vector `BV.CHRIX`, which points to one of a sequence of memory-allocation routines, and adds an offset to manipulate the channel table routine instead of the `RI` stack one. This offset of 52 is guaranteed on all Qdos systems, as Tony Tebby uses it, and has told Laurence Reeves and David Oliver about it; Amiga Qdos developer Mark J Swift can be relied on not to change things like that.

Keep it tidy!

The new 40-byte entry could contain anything when you get it, so the program stores the

channel ID followed by 31 zero bytes which reset the print and graphics positions. The odd byte holds 80, the default `WIDTH`; if this is zero 'division by zero' exceptions will occur when you use fancy separators like `"TO"`, `"."` and `"!"` to `PRINT` through the channel anywhere but the screen.

The main challenge in `INPUT$` is keeping the `RI` stack tidy. Thankfully you do not have to worry about this in resident procedures and functions that fail, but functions that return without error `MUST` ensure that the result is the only thing on the `RI` stack when they've finished. The `RI` stack is often held in `A1`, but the official value is `BV.RIP`, a long word at `$58(a6)` in the SuperBasic memory area. Use `BPEEK.L(88)` to read this from SuperBasic.

INPUT\$ reads two integer parameters onto the stack, then checks them, before allocating enough extra space for the text and a prefix length, on an even word boundary. It calls BV.CHRIX vector to check there's enough room, then uses TRAP #3 and IO.FSTRG to read the string direct from the device to SuperBasic memory. The TRAP #4 warns Qdos that the string address is relative to the start of SuperBasic.

SET_POS uses just twelve instructions, plus a call to the subroutine it shares with INBYTES, GET_QDOS_ID. This

attempts to convert an integer SuperBasic channel number in D0 into a channel ID in A0. If anything goes wrong it scraps the last return address with ADDQ.L #4,A7 and returns error code -6 (channel not open) directly to Basic, so there is no need for calls to check D0 on return.

SET_POS reads its parameters with a single call for two long integers. It saves time and space if you let Qdos coerce the integer channel number to a long word, rather than make two vectored calls: one for the integer and another for the long word.

TABLE - ANYOPEN% report codes

- >=0: new SuperBasic channel number available for use
- 3: out of memory - are you still running a 128K QL?
- 6: no room for a new channel; over 360 open (or 168 in 128K).
- 7: the device name was not recognised
- 8: the file already exists (after open-type 2)
- 9: the file or device is already exclusively opened
- 11: there's no room for the new file on the disk
- 12: faulty device parameters, or too long a file-name.
- 16: a euphemism for the dreaded 'bad or changed medium'
- 20: read-only - the drive or file is write-protected

Telephone: 0753-888866
Fax: 0753-887149

(EEC) **W.N. Richardson & Co.**

SINCLAIR QL PRICE LIST MARCH '93

18-21, Minbourne House,
Chalfont St Peter, SLB GLE

All Prices Include 17.5% VAT

QL Computers NEW MONITOR £50 CHEAPER

IS COMPLETE QL Computer, PSU, T.V. Lead, PSION V2.35 Software (Word Processor, Spreadsheet, Database & Business Graphics), Handbook for QL, Programs, & Superbasic. JS £120
Free one year membership to QUANTA, the independent QL User Group with over 400 free library programs, Newsletters & H.E.P. 6 MONTHS WARRANTY. WITH 1M ROM £100.00

PART EXCHANGE £30 OFF ABOVE. Send in old QL (Unit only, any condition) - JS ROM £90 JM £75.00

BACKUP QLS QL & PSU only. JS £80 JM £65 (Part exchange allowance £15 if required)

Accessories

* NOTE: EXTERNAL (SER2) 3 BUTTON MOUSE AND SOFTWARE WITH EXTRA FUNCTIONS. NOW HERMES COMPATIBLE *

PC KEYBOARD INTERFACE, INTERNAL FITTING, FOR FITTING 102 KEY KEYBOARD £75.00
PC KEYBOARD, UK version, 102 key (AT) £30.00
PC KEYBOARD INTERFACE AND KEYBOARD (INCLUDES FREE JOYSTICK OR PSU) £95.00
CASE AND LEAD FOR EXTERNAL FITTING of Keyboard Interface £14.00
JOYSTICK with QL lead (no interface required) £5.00
* QL MOUSE, 3 Button, Software controlled, externally mounted, simply fits in SER2 £45 CORDLESS MOUSE £55.00

Disk Drives

THE UNIVERSAL DRIVES ARE ALSO SUITABLE FOR ACORN BBC, ATARI ST, AMIGA, SPECTRUMS, IBM, AMSTRAD AND OTHER PC COMPATIBLES

Universal 3.5" 1Mb disk drive, cased with PSU and free QL lead £120.00
Universal 3.5" 2Mb disk drive, as above but 2Mb capacity £90.00 2 Units for dual drive £170.00
3.5" 1Mb Uncased disk drive £25.00
3.5" 2Mb Uncased disk drive £25.00
Lead for uncased disk drives, or other universal micros £10.00 POWER SUPPLY FOR UNCASSED DRIVES £6.00
METAL CASES FOR UNCASSED DRIVES £6.00

Parallel Printers

NEW RANGE

SAMSUNG SP093 80 Col, 300 cps, 50 NLQ, 3K Print Buffer, Centronics, Tractor Feed/Sheet Feed, Paper Park £149.00
OLIVETTI COLOUR PRINTER, 24 pin, 200 cps, 50 NLQ, 40K Buffer, Epson LQ2550/IBM Compatible £259.00
CENTRONICS PARALLEL Printer Interface for above printers £27.00
QL LEAD FOR SERIAL PRINTERS £10 CANNON BJ10 EX £225

Monitors

PHILIPS 14" HIGH RES COLOUR EGA .31 DOT PITCH, FULL 85 COL WITH AMBER OR GREEN TEXT FEATURE, IDEAL FOR WORD PROCESSING, RECONDITIONED, 90 DAY WARRANTY. £149
OK FOR PC AND QL SELF SENSING AND MANY EXT'L CONTROLS. £39

Microdrive Cartridges and Spares

4 New Cartridges in a wallet £10.00 8 prog carts for reformatting in 2 wallets £15.00
Pastic Storage filing box including 20 new cartridges £45.00
QL Pison Software V2.35 Includes Quill, Abacus, Archive, and Easel IN WALLET £18.00
QL Pison Software Separate programs £10.00
QL power Supply Unit £10.00 Membrane (and instructions) £9.00
TV or Network leads £3.00 QL Top & Bottom Case £5.00
IC's ZX £301 £6.00 ZX £302 £3.00 8049 (IPC) £3.00 MC 1377 £5.00
MC 68008 £12.00
QL SERVICE MANUALS & CUTOUT £25.00 S.A.E. FOR FURTHER DETAILS

Payment terms:
CWO, Access, VISA at cetera
Cheques - allow 10 days
Product subject to availability, E&OE
TEL: 0753 888866 FAX: 0753 887149

Delivery:
Carriage - £9
Postage - £5

TF SERVICES

MINERVA

The ULTIMATE operating system upgrade

MKII MINERVA with battery for 256 bytes ram, CRASHPROOF clock & Philips PC bus for interfacing. Can autoboot from battery backed ram. Quick start-up.

Other features common to MKI/MKII
DEBUGGED operating system/ autoboot on reset/
Multiple Basic/ faster scheduler-graphics (10% of
lighting)-string handling/ WHEN ERROR/2nd screen/
TRACE/ foreign keyboard drivers/ 'warm' fast reset/
Auto reboot after power failure. V1.97 now with built in
Multibasic and split OUTPUT baud rates with Hermes.

1st upgrade: free. Otherwise £3 (+ £5 for manual if req'd)
(send sat, rom & NEW disk/3 mds)
MKI to MKII upgrade - £30

MKI.....£40 MKII.....£65

BOTH VERSIONS GOLD CARD COMPATIBLE

HERMES

A replacement QL co-processor for the QLS awful IPC 8049

- Do you get keyboard bounce?
- Do you find fast serial input unreliable?
- Do you want to connect a modem at 19200bps

If you can say YES to any of these, then you need HERMES

- 19200bps RELIABLE serial input - NO QCONNECT.
- Independent input baud rates - use serial mouse & print
- Stops keyboard bounce (unwanted repeat chrs)
- Improves 'fuzzy' and 'random' sound
- Provides extra input/output lines
- Key click

Fitting is simple. Remove the QL top (8 screws) & replace the chip marked 8049 or 8749 next to mdv 1.

£25 including manual/software

QL SPARES

Faulty QL board (no plug-in chips).....£9

Keyboard membrane.....£9 Circuit diagrams.....£3
68008 cpu.....£8 1377 PAL.....£2
1M ROM.....£10 Power supply.....£12
8302 ULA.....£10 8301 ULA.....£10
8049 IPC.....£8 MDV ULA.....£12

Other components/sockets etc) please phone

QL REPAIRS

Fixed price for unmodified QLs, excluding microdrives.
QLs tested with Thom-EMI rig and ROM software

£27 including 6 month guarantee

All prices include post & packing (UK only). Payment by Mastercard/Visa/Burocard/cheque/postal order/PO Giro transfer (58 267 3909). MAIL ORDER ONLY - no callers without ringing first. Ring for overseas prices.



12 Bouverie Place, LONDON W2 1RB

Tel: 071-724 9053 Fax: 071-706 2379



QUANTA

Independent QL/Thor Users Group

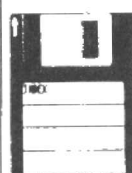
Worldwide Membership is by subscription only, and offers the following benefits:

- Monthly Newsletter - up to 40 pages
- Massive Software Library - mostly FREE!
- Free Helpline and Workshops
- Regional Sub-Groups. One near you?
- Advice on Software & Hardware problems
- Discounts from most major suppliers
- Subscription just £14 for UK members
- Overseas subscription £17

Barclaycard: Visa: Access: Mastercard

* Now in our NINTH successful year *

Further details from the Membership Secretary



Bill Newell
213 Manor Road
Benfleet
Essex. SS7 4JD
Tel (0268) 754 407



SJPD Identified

SJPD **public domain software** and shareware have added the following new disks to their library.

SJS 60 - a disk of text files discussing UFO phenomena; SJS 61 **Xtricator** 1.40 ZX81 emulator for the QL by Carlo Delhez, along with some ZX81 public domain software; SJS 62 **DBAS**, a programmable database replacement for Archive (it says here), by David Howells; SJS 63 **Graphiste**, a drawing program from France, translated into English for SJPD Software by Tony Burns.

The following disks have been updated: SJS9 **Molecular Graphics** V5.11, now Gold Card compatible; SJS 52 **DJW demo** disk.

All SJS disks are £1.75 inclusive of media and return postage, or £1.00 copy charge where user supplies medium and adequate postage.

SJPD also has a list of **PC shareware** under the PCPD label, including PCPD 2 QL-PC, two utilities, one to aid transfer of QL files to the PC, and one to aid the PC to view QL screens. PCPD disks are available on 3.5-in disks only, and are £1.75 each fully inclusive.

You can get SJPD's catalogue either in printed form (4 x 24p stamps) or fully up to date on disk in return for a formatted disk, postage and adhesive address label.

SJPD, 36 Eldwick St., Burnley, Lancs BB10 3DZ.

Review rises again

QReview is the new QL magazine by Bruch Nicholls of **Quo Vadis Design**. QReview is the successor to QL Technical Review and QL Leisure Review, the periodicals that Bruce used to produce for CGH Services.

Following very much in the style of its predecessors, QReview has a look at data transfer between computers, Printmaster, Super Disk Labeller, Adventure Playtime and the Pointer Environment. There is also a report from the International QL Meeting in Eindhoven in February, and some other items, including a list of the best-known QL suppliers, a news page, a questionnaire and order form for previously issued of QLTR and QLLR, and a competition for programmers to win a copy of Perfection.

With 36 A4 pages of clear print, QReview Volume 1 Issue one is going for £2 in the UK, £2.50 in the rest of Europe, and £3.00 elsewhere. **Contact Bruce Nicholls, 57 Shaftesbury Road, Romford, Essex RM1 2QL.**

All is not equal

In the 11.7 issue of **QL World**, and to a smaller extent in earlier issues, in some places the "equals sign" in equations has printed out as a "fat minus sign" in the text. In most cases it is fairly clear that this is a print fault, but if you are confused by a "minus" term, or find that it does not run, and the "minus" looks unusually black, try replacing the "minus" with an "equals" sign. We'll get rid of the fault as soon as we can.

Quanta Raynes, OK

The Quanta **Essex Workshop** will be held at the Rayne Village Hall (opposite the Swan pub), Gore Road, Rayne, Braintree, Essex, from 10 am to 6pm, on Sunday 19 September. All QL users welcome, say Quanta: refreshments, demonstrations, talks. Contact **0376 347852** for more information.

Mersey Man Dies

After the **Mersey Mouse** was mentioned in the June QL World, Ron Watson of the QL Mersey User Group contacted us with current details of the Mersey Mouse.

Sadly, since then QLMUG secretary and mouse team-leader Don James has died. "Don, who was a founder member and Secretary of the QL Mersey User Group, will be greatly missed by his wife Bett and family, and all of his friends in the QL world, especially those in the QLMUG," writes Ron. Don spoke to us from time to time about QLMUG and the Mersey Mouse, and was a real enthusiast and pillar of his community.

It is too soon as yet for a new Secretary to be chosen, or for a new sales address for the Mouse, but chairman Peter Tyler can pass on information about the Mersey Mouse. The pack at present is £37.50 for a mouse and interface complete and ready to plug in, £25 for a complete kit with full wiring instructions, or £6.50 for a set of components, £6.00 for the pcb and the mouse itself for £12.50. An information leaflet is also available. **Contact Peter at 26 Ryder Crescent, Aughton, Nr. Ormskirk, Lancs. L39 5EZ.**

QLS
WORLD

QUICK-LASER 1.03

Bryan Davies makes the most of his laser printer.

INFORMATION

Program: QuickLaser 1.03

Price: £19.95

Supplier: Digital Precision Ltd.,
222 The Avenue, Chingford,
London E4 9SE. Tel. 081 527
5493.

The quality of desktop publishing documents was, for many years, severely restricted by the coarseness of dot-matrix printing. You could not expect much from a print head with nine needles, even when the effective number was increased by various enhancement techniques. The situation was much improved by the advent of 24-pin printers at a reasonable price, but it was still obvious that print was from a "dotty" machine.

All that changed radically when laser printers arrived, and rapidly came down in price to a

level that a fair number of average users could contemplate. Ink-jet printers then offered print quality comparable to that of lasers, for less money.

Driver shortage

What did not occur on the QL scene, for quite a while, was the development of printer drivers that could send DTP output to laser and inkjet printers. There was little point claiming the DTP program itself had capabilities similar to those of much more expensive products when the printed results looked markedly inferior. A link was obviously missing - until QuickLaser came onto the market.

Clearly, QuickLaser ("QLQL" from here on) was produced for use only with its stablemate,

Professional Publisher. As for laser printers, it should work with any of them that claim compatibility with the widely popular Hewlett Packard LaserJet Series II. Inkjet or other types of printer should be suitable, also, provided the magic "HP LaserJet Series II compatible" item is in their specifications. But be careful with that description, because HP has a rather confusing range of model designations, with various models with "II" in their designations. What the compatibility claim refers to is the "Series II", one of the earlier LaserJets. Most laser printers made by manufacturers other than HP have an emulation mode which fits the bill, but check the specification for the right words - anything without HPLJII compatibility can be close to useless because it will

not be usable with much software. Digital Precision point out that (obviously) they cannot accept responsibility for failure of a particular printer to behave like a LaserJet.

QLQL prints in graphics mode, and uses the full 300 x 300 dots per inch resolution of the LaserJet. The results are good. How good, and how fast, depend upon the ram available in your QL system, and the page dimensions of the Professional Publisher page being printed. Some distortion will occur with certain combinations of page size and print mode.

The program comes on one 3.5-inch DD disk. It requires a minimum of 384 kB system memory to run. Performance is stated to be appreciably better with the 896 KB or more of the full Trump Card or the Gold Card. Both serial and parallel printer ports are usable. The output of the basic QL is always serial, of course. Parallel output is available from a few, older interfaces, but the usual method of driving a parallel printer is through a Miracle serial-to-parallel converter. The input devices supported are the usual flp, mdv, fdk, ram, win, and network device names (from SuperToolkit II).

SuperB extensions

SuperBasic extensions files - three of them - are supplied, and are loaded automatically by the boot routine. You can "multi-task" QLQL, and the obvious program to run alongside it is Professional Publisher. A better phrase is "task switch", because you are told in the instructions that ProPub should not be used while QLQL is printing. During sessions when other programs are active at the same time, the

DIGITAL PRECISION BRINGS YOU ROCK SOLID SOFTWARE!

Professional Publisher is simply the best. No other desktop publishing program for the QL even comes close!

A new standard of printing!

Professional Publisher has always been regarded by those who knew it as a superb program, giving the best results possible from dot-matrix printers. Now there is "QL QuickLaser", the printer driver that allows Professional Publisher to output at the full 300 dots-per-inch of LaserJet printers! QL QuickLaser (QLQL for short) will drive any Hewlett-Packard LaserJet II printer, or any true compatible. If you have only a 640k RAM QL, then you may still use QLQL to drive one of these fine printers, but at an acceptable 150 dots-per-inch, or at 150 by 100 dots-per-inch.

disk, and QLQL was used to dump the resulting file to the printer, a Hewlett-Packard LaserJet II. If you have enough RAM, you can have both Professional Publisher and QLQL in memory together, speeding things up tremendously.

though, only needs to have the Trump Card 768k to utilise the full 300 dots-per-inch that you see on this page! Look at the examples of text, fill patterns and greyscales in the sample box. All of the fonts on this page are either shipped with Professional

"Roman" text. AaBbCcDdEeFfGgHhIiJjKkLl
"Syreeta" text. AaBbCcDdEeFfGgHhIiJjKkLl
"Louise" text. AaBbCcDdEeFfGgHhIiJjKkLl
Greyscales as possible! Pattern
fills are available, too. 
Italics can be made from any font!

QLQL screen may become corrupted; this is easily fixed by pressing F4.

To show off QLQL's paces, a sample ProPub page is provided, which is 1920 x 1600 pixels in size. This size requires 896 KB or more of ram, if it is to be loaded into ProPub. Another file supplied is named FILL_PATTERNS, to replace the existing file of that name that comes with ProPub. The new version gives better grey-scale and pattern-fill printing with laser printers. If those terms do not mean anything to you, there is still a lot for you to learn about DTP!

Images are not in pure black and white, unless they are so-called line drawings, or text. An obvious line drawing is a pencil-drawn square; the box lines are black, the rest of the area is white. You may have seen the drawing of the Space Shuttle Columbus, which is a common sample file with many drawing and DTP packages; that too is a line drawing. To give depth to an image, levels of blackness have to be present, and the individual levels are part of the "grey

Fill patterns

Fill patterns are used to add "body" to objects which would otherwise look white with black lines around them. For example, there are fancy text fonts with ProPub that use "hollow" strokes for characters. The inner parts of the strokes can be decorated with fill patterns, to make the text look more interesting. This effect is demonstrated clearly by the sample page supplied with the program - see Figure one.

The instructions provided with QLQL cover 10 pages. The first few pages contain helpful notes on the practicalities of printing images through laser printers. Users who have spent years struggling to get decent printing from dot-matrix printers may well be "clueless" when it comes to using laser printers, so the advice is welcome.

There may be an UPDATES.DOC Quill file, containing additions to the instructions; none was present with the review copy. You are advised that DP's Perfection word pro-

plied with Perfection SE should be used to maintain formatting (text attributes etc.)

Laser limits

Users unfamiliar with laser printers may be surprised to find them sometimes unable to print graphic images. The basic LaserJet has 512 KB of memory, which might be thought adequate for anything, in view of the fact that dmp printers usually have not more than a few KB of memory. The laser method of printing is significantly different from the dmp method, however. One difference that should be obvious from the first print you make with a laser is that the complete page is printed at one go, rather than done strip-by-strip, dmp fashion. When you think about it, that is likely to mean the laser has to store the data for the whole of the page, before it can start printing it. This is where the amount of printer memory becomes important.

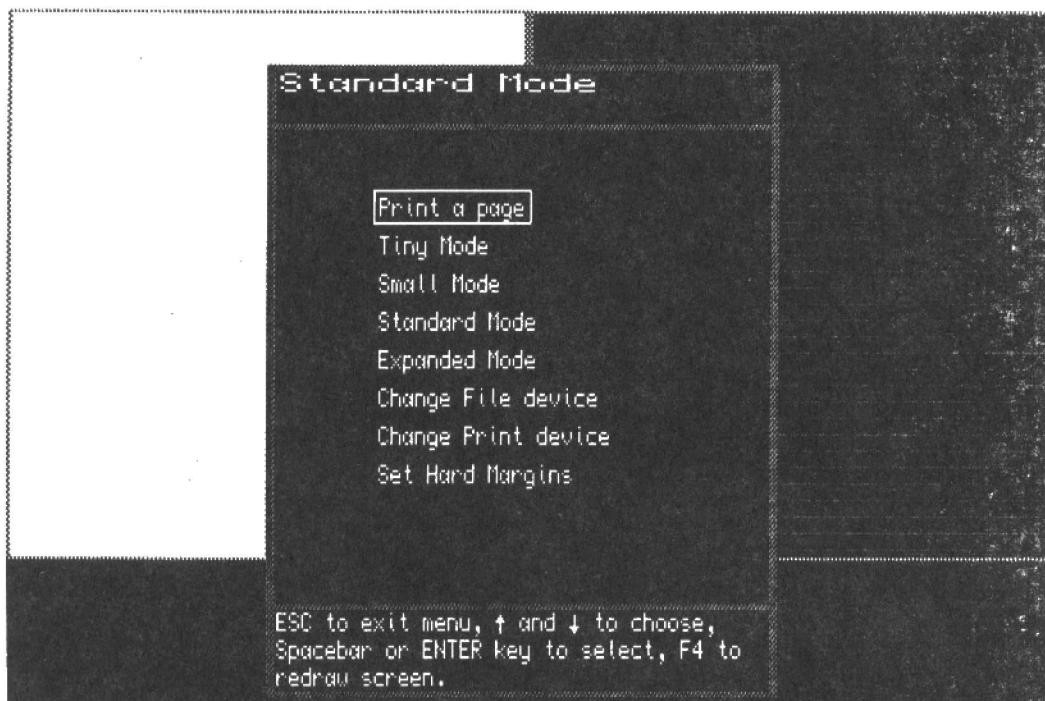
The instructions state that a LaserJet with the standard 512

graphics to a HPLJII with 512 KB, but they are the exception rather than the rule. Should you be stuck with the basic 512 KB - and most people with HPLJIs that I have met are - you can print with QLQL, but the process involves putting the paper back in after the first run, and it takes much longer than does printing with 1 MB or more. Another drawback is that some mis-registration will be apparent, where the two parts of the image meet.

Graphics and text

Impatience is something that has to be suppressed, when printing graphics. The process is much slower than text printing. In the instructions, comparative times are quoted, for the various printing modes of QLQL. The figures are 15, 16, 14 and 210 seconds, for the Tiny, Small, Standard and Expanded modes, respectively. The operation they refer to is the "printing" of a 960 x 800 pixels test page from one file on ramdisk to another one. That is, this is the least time that could be taken, on a standard Gold Card system. Adding the printer into the chain makes the time to print the same page about 15 minutes with the first three modes, and about twice that long with Expanded mode.

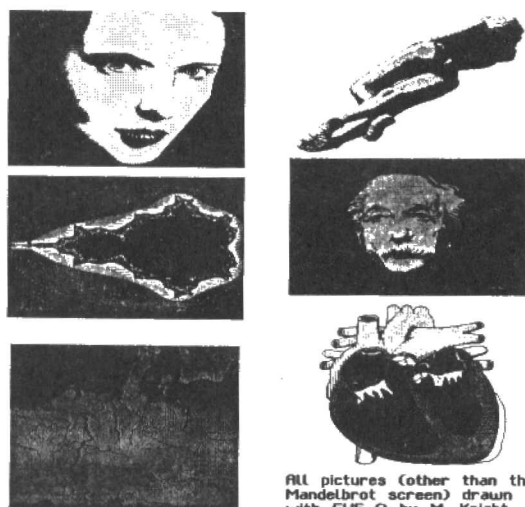
My experience, printing to the Epson GQ-5000 set to HPLJII emulation mode, was both better and worse than the quoted figures for the LaserJet, but not sufficiently so to get excited about. The sample page took under 13 minutes, from floppy disk to printer, using the Small mode; using Expanded mode, the same page took an hour. The Expanded print was halted by a "buffer full" error message from the printer, about half-way through the time, and the paper had to be re-inserted to get the last bit of the image printed. The Epson does have 2 MB of ram, but it is not clear what influence this has on speed, or on the size of image that can be coped with. A useful feature that could be added to QLQL is some form of progress indicator, to let the waiting user know whether he



scale". Colour images cannot be reproduced by most printers, and the different colours need to be represented by suitable grey scales, in order for the printed image to look reasonable.

processing program is the best way to prepare text for use in ProPub, but that does not stop you using other WP programs for this job. ProPub itself can do any required micro-justification of text prepared with Perfection. The PUBCONVERT utility sup-

KB of ram will not print a full A4 page of graphics. In practice, this is what appears to be true. You will need 1 MB of ram for graphics printing from many programs, including QLQL. To be strictly correct, there are programs that can print full-page



All pictures (other than the Mandelbrot screen) drawn with EYE-Q by M. Knight.

or she has time to go away and do something else.

Multiple gain

One feature of laser printing that should be borne in mind is the difference in time taken for several copies, compared to one print. The bulk of the time is taken in the production of the first print; subsequent prints should be done very much faster. This applies only if the number of copies is set at the printer. QLQL itself does not have a multiple-copy setting.

The initial menu is shown in Figure two. The highlight box is on the "Print a page" option, and you can simply press Enter with it there, select a file from the list that is presented, press Enter again, then go away and make coffee while the program and printer think it over. The default printing mode is Standard. From the sample 1920 x 1600 pixels page, that produces an image filling an A4 page (in portrait orientation) but showing only half of the original page image. Selecting the Tiny mode before printing results in the whole image being printed, covering half of the portrait page. Small mode printed the whole image, taking about three-quarters of the page for it. That leaves just Expanded mode, and that pro-

duced a full-page print, of one quarter of the original image. The quality is excellent in all modes.

It is the Small mode which was used to produce Figure one. QL World now has a superior method of transferring images to the magazine pages, and the detail in the fancy text should be reproduced quite well. Look at the fill patterns in the text box, and at "ROCK SOLID" in the heading. Those with vivid imaginations can now let them run riot. It is not even necessary to design your own founts, as some good ones come with ProPub. For contrast, Figure three contains several drawings, done by the writer of QuickLaser. He used DP's Eye-Q graphics program to produce this page. The print was done in Standard mode, and the quality speaks for itself.

Disk tactics

Once QLQL has been started, you can remove the program disk and insert a disk containing images. Alternatively, you can load ProPub from another disk. This is where the advantage of having ED drives really becomes apparent. QLQL opens the file to be printed, and keeps it open throughout the process. To save time, and take

less chances with image files, it is preferable to copy the latter to ramdisk. One of the perennial problems with printing is not being able to stop it when something is seen to be going wrong; the QLQL instructions offer a primitive, but effective, way of dealing with this - pull the disk out! The error condition is then noticed much faster by the program than it will be after the Esc key is pressed.

The image page sizes advised for a 640 KB system are 960 x 800 or 960 x 960 pixels, for Expanded and Standard mode respectively. The print resolution is 150 x 150 dots per inch - a quarter of the maximum capability of the LaserJet. To get more, you need more memory in the QL system. When the ram size is 896 KB or greater, use the page size 1920 x 1600 pixels and print from the Small mode. This makes use of the full 300 x 300 dpi available from the printer, giving the best results. The Tiny mode is recommended for printing letterheads.

Dottiness of text was not evident in the Tiny and Small print modes, but began to show with Standard, and was fairly evident at Expanded. Bear in mind, though, that this assessment is based on years of experience with laser printing by dmp standards, none of the printed samples would be considered dotty. One warning is given towards the end of the instructions, concerning printing saved partial pages; the width of such pages must be a multiple of 8 pixels. The height can be anything from 1 to 1600 pixels.

Conclusion

The facility to use a laser printer is essential for good quality DTP output. This program is very easy to use; definitely not in the head-scratching category. When you have spent the money for a laser printer, the additional cost of QuickLaser is small, and the improvement in output over dot matrix print is large. Add a little for extra coffee, though, while you wait for the lovely pictures to emerge.

QUICKLASER 1.03

QL Bulletin Boards

Hilary Snaden recommends BBSs as a source of software and QL contact.

Most QL owners looking for free software will probably look first to public domain libraries, but those with modems and terminal software have a further option. Bulletin boards are well-known as an electronic messaging facility, but there is also a good deal of free software to be found on them. Much of this is not available elsewhere, having been put there by programmers for trial and comment or in response to another board user.

This article attempts to give a few hints to those who have not explored this part of the computer scene, and to highlight some of the best software on QL bulletin boards.

XMODEM

To be sure of being able to download software from a QL bulletin board you will need (of course) a modem, and terminal software with the *Xmodem* file transfer protocol built into it.

It is possible to download basic programs and text files without Xmodem, by using the View option on the bulletin board menu to display them on-screen while simultaneously logging them to disc, and then correcting (if possible) in a text editor any errors which may have crept in due to line noise. However, this will almost certainly fail with machine code, as errors will be more difficult to detect, and because raw machine code will frequently contain control codes which will be intercepted by the receiving

software.

Xmodem works by sending a few bytes at a time and then making sure that a checksum in the data corroborates that they have been received accurately. This makes it slightly slower than downloading data unchecked, but it is more than worth the extra time. If, however, you have a noisy line then even this may fail, as Xmodem will only resend a block of data so many times before aborting.

Charges

The only cost to be incurred in downloading software from bulletin boards is that of the call charges, so the faster your modem, the less you will have to pay the telephone company. But make sure that your terminal software is set for a speed compatible with the bulletin board you are trying to log on to.

To find out exactly how the file transfer options work on your terminal software, read the manual. Some aspects of file transfer, particularly causes of failure, are not obvious but useful to know, and these may well be explained. As a rough guide to timing with Xmodem, a 50K file will take around ten minutes to transfer at 1200 baud, proportionately more or less at lower or higher speeds.

Many of the files on bulletin boards are compacted or archived to save space on the host system, and time and money for the caller. In all cases the appropriate decompacting or unarchiving software is avail-

able from the same boards, so plan to download these first.

The most archiving common methods used on QL-based boards are CPT (used for single files), ARC and ZIP. ARC and ZIP are the same as on MS-DOS and other machines, and can handle multiple files, so that a group of related programs, documentation, etc., is compacted into a single file with the suffix `_ARC` or `_ZIP`. These need the programs called UNOPT, ARC and UNZIP to get at the encoded files.

You may also encounter the suffix `_LZH`. This signifies a more recent archiving method requiring a program called QLHARC to access the data.

Unarchiving

The unarchiving program will extract all the component files at the same time as decompacting them, as well as ensuring that EXECable files are written to disk with the essential file header information which tells Qdos how much dataspace to allocate to the program.

If you intend to frequent bulletin boards it may be useful to keep all these file "hacking" programs on a single disk, along with the terminal software. Remember also that both ARC and ZIP have a number of options available, such as listing the contents of an archived file without extracting it, and it is not always obvious how to use them, so read the documentation, and the advice on downloading given on the boards

themselves.

You will need to leave your name (your real name), and telephone number when you first log on, so that the sysops (system operators) know who you are. It is unlikely that you will be able simply to pick a convenient bulletin board, log on and immediately start downloading software, as most sysops will wait a while for users to have logged-on a few times before giving them access to the files.

Contribute!

This is by way of a subtle hint that bulletin boards are a two-way communication system, and users are encouraged to contribute to it. You may not be much of a programmer, but you will almost certainly have some knowledge which could be of use to fellow board users, so rather than heading straight for the files area, have a look around the message area and see if you can help with the problems or contribute to the debates which are the hallmark of a lively board.

It is often the message area rather than the files which justify the existence of a bulletin board, and if no-one bothers to keep it alive it will close down and no-one will have the benefit.

You will also be able to upload software and other files to the bulletin boards, and are positively encouraged to do so. In this, users limited to V23 (1200/75 baud) modems such as the Qconnect will find themselves at a serious disadvantage, since

while receiving takes place at 1200 baud, sending at 75 baud is very slow. In this case it may well be cheaper to send the files on a disk by post to the sysop with a request to place them on the system. Note that any software uploaded onto bulletin boards must be either original or placed with the permission of the copyright holder (usually the author, but sometimes, as with some ex-commercial software, a publisher).

Closing down

As with proper computing practice everywhere, it is important when logging off to use the proper command on the menu of the system you are connected to, rather than simply dropping the line, a guaranteed method of incurring the enmity of the most mild-mannered sysop should you make a habit of it. Using the proper command keeps the host system in touch with what's going on, and ensures that the file and message pointers which it maintains for your benefit are updated. Dropping the line should only be used as a last resort, for example if the host system has locked up, as can occasionally happen.

My list of interesting software is by no means exhaustive, merely giving an idea of what can be found, and there are a great many other programs, from games to graphics demonstrations, which users may find useful or interesting.

Different bulletin boards may have different versions of the same program, depending on how much time the sysop can spare for keeping the file section up-to-date. Where these are known, the latest version is described as this will almost certainly be an improvement on its forebears.

It is difficult in a short space to do justice to Jonathan Hudson's *QeM*, which has by August reached version 3.6. While *QeM* cannot handle Viewdata format boards such as Prestel, it has in addition to the XMODEM file-transfer protocol mentioned earlier, YMODEM, ZMODEM, and KERMIT protocols, and VT52, VT100 and a number of alterna-

tive ANSI emulations.

The terminal emulators will run in an unexpanded QL, but extra memory is required to make use of the external file transfer protocols. Since the program has neither a permanently active cursor, nor a screen restore routine, multi-tasking is best done with the help of the Pointer Interface and Window Manager.

Considerable configuration of the program is possible, and although the suggested method for use with the Tandata QConnect-QMod_QCon modem does not appear to work, Hayes-compatible modems can now be found cheaply, and *QeM* seems more than happy with them with or without the assistance of the QConnect RS232 port.

It is not an easy program to set up, particularly for newcomers to comms, but it is worth the trouble. If you have a QL and a Hayes modem, *QeM* is the program to start a deep and meaningful relationship between them.

Qdos to MS-DOS

IBMDisk is another utility by Jonathan Hudson which, like the better-known *DiscOver*, provides directory information on and transfers files between Qdos, MS-DOS and Atari ST disks. It is well-documented and delightfully easy to use. Look for version 1.9. This program requires a minimum of 256K ram, two disk drives and Toolkit II.

RESQL, by Hans Lub, may provide an escape for those awful moments (we have all done it) when we realise, in the moment that our finger presses Enter, that we didn't really want to delete that file at all. If the disk has not been written to since the unwanted DELETE, *RESQL* can map the sectors which the delete operation has marked as "free", and "undelete" the file by piecing together the fragments and writing the reconstructed file to another disk. The version number of 0.00 suggests that this is a prototype, and the "New disk" command seems occasionally to lock up the program (though if this happens one can

still remove the program, using Toolkit II's RJOB command, and start again) but it is nonetheless very useful. Minimum requirements are 256K ram, a disk drive and Toolkit II.

QED is a superb piece of programming by Jan Bredenbeek, author of versions of the UNARC and UNZIP utilities found on bulletin boards, as well as QBOX and HARDBAK. It is a fast text editor with comprehensive documentation, on-line help and a configuration program. *QED* works something like Digital Precision's well-known The Editor, though without the ability to handle binary files, but is less than 8K in length, so multiple copies can be run even in unexpanded machines. It multi-tasks immaculately using EXEC, though it is also an excellent candidate for installation under a basic keyword, and is highly recommended.

Your own BBS

QBOX is the host bulletin board software used by most QL-based systems. If you fancy trying to set up your own board, you can download the program, documentation and other associated files for perusal. *QBOX* is shareware rather than public domain, and if you intend using it seriously you will need to pay a modest fee to the author to be kept up-to-date with the latest developments, and in fairness for his many hours of programming time in developing the program.

HARDBAK is a simple and very compact program intended primarily as a means of backing-up the contents of a hard disk to a string of floppies, but it can also copy files between any two directory devices. Unlike many copying utilities *HARDBAK* preserves the source files' original timestamps, and since it makes use of as much free ram as it can find (less 16K) the copying process can be very fast indeed with this program.

Since the first, somewhat tentative, days of QL-based bulletin boards a number, in both the UK and Europe, have become linked to a worldwide network of computers called Fidonet. Messages are exchanged

between the various nodes on the network at less busy times and when long-distance calls are cheapest (around 3 am); from the QL user's point of view a number of message areas are in effect shared, and messages can be sent to or read from any of the boards on the network from the one nearest and cheapest to call. This facility does not generally apply to files, however, and if the file that you want is only available from one particular board then that is the board you will have to call.

Boards to try

The bulletin boards mentioned here also have message and file areas dedicated specifically to programming in C, and often have the latest versions of components of the increasingly-popular C68 compiler before they are available from public domain libraries. They are also the most convenient way to find information on how to get the best from the Minerva operating system.

The following UK boards have been found to have particularly good selections of software available, and being very reliable. However, most boards aimed at the QL community have a range of software, and all are worth calling, particularly as the bulletin board scene is always changing, with new software appearing and, occasionally, new boards being started.

Changes of telephone number also occur from time to time and the latest information on all of this can usually be found on other boards - it pays to keep in fairly regular contact.

Both require calling software to be set to scrolling mode, space parity, and one of the following speeds: 300/300, 1200/1200, 2400/2400 or 1200/75.

QBBS

Sysop: Tony Firshman
Telephone: 0344 890987
Hours: All day

4th dimension

Sysop: Wayne Weedon
Telephone: 0202 600305
Hours: 9am - 6pm only

SuperBasic In Action

Simon Goodwin builds a comprehensive file-requester in extended SuperBasic.

A file requester lets you select files on any device by pointing at names in a list. **Requesters** are so useful for file manipulation that they have spawned full-blown utilities like Xtree for PCs and SID or DirWork on the Amiga.

These utilities are little more than multiple file requesters, with a few 'buttons' to perform functions like copying, viewing or printing files, but they're so easy to use that they're often preferable to a QL-style command line, especially when browsing or re-organising data on disks.

DIY Toolkit enthusiasts will remember the `_DEF` functions for SuperBasic, which let you select procedure and function names from a menu. The SuperBasic file requester is similar, in that it displays a list of names which you can scroll through to make a selection, but it has many more features.

Multiple selections are allowed, so you can pick any batch of files at one stroke. A 'scroll-bar' shows the proportion of the list on the screen - unlike the PC Windows equivalent - and your position in the list, updated as you move through the names.

The **SuperBasic in Action** file requester reads all available device names from system memory and lets you pick any drive by name, like a file. It also supports sub-directories on systems that organise files that way. The default suits QJump's level 2 drivers, as used on the Gold Card and upgraded ST emulators, Trump Cards and SuperQBoards; it should also suit CST Thor sub-directories if you change the file type it is looking for from 255 to 3.

The program works fine if you

do not use sub-directories, and handles up to 480 files per disk - the theoretical upper limit for a 720K drive. You can increase this if you've got a Gold Card, and memory to burn; the default uses about 20K for a full directory. The limits are encoded as strings in the program so you can still patch them easily after compilation.

The requester works in **Mode4**, **Mode8** or **Mode12**, showing full Qdos file names with no restriction on length. You can configure the character and window-sizes, colours and border, and fit two full-sized 25 line requesters side-by-side on a Mode4 monitor screen.

Mice and sticks

Selections can be made with the cursor keys, or a joystick in port CTRL1, or a mouse with keyboard emulation, like SERmouse or the Amiga Qdos mouse driver. Most two or three-button mice are suitable; if it works at all with Quill, it should work well

with the SuperBasic in Action file requester.

You may need to press Shift or Alt as well to select short-cut functions, like paging, drag selection and direct moves to

the start or end of the list. It uses Enter and SPACE to 'do' or select entries, and Esc or the middle button on a three-button mouse to escape, as usual.

The program makes abundant

```

100 REMark SuperBASIC in Action QDOS FILE REQUESTER
110 REMark By Simon N Goodwin for Sinclair QL World
120 REMark Version 1.4, tested 3rd-17th August 1993
130 REMark Extensions: INPUT$, EDIT$ or EDLINE$, CHAN_W$, ANYOPEN$,
140 REMark PEEK$, SYSBASE, NEWCHAN$, MINIMUM$, STRING$, SET_PRIORITY
150 REMark Two TURBO compiler optimisation directives follow:
160 IMPLICIT% i,j,k,n,d
170 DATA_AREA 20
180 :
190 path$="flp1_" : mark%=0 : MODE 4 :REMark or 8 or 12
200 PRINT #0:"At '" & REQUEST$(#0) & "' in '" & path$ & "'"
210 :
220 DEFine FuNction REQUEST$(status%)
230 REMark STATUS% = report window # (2 by 36 chars or more)
240 REMark Globals: MARK% = select default; PATH$ = drive/directory
250 REMark Bit 8 of FLAG% entries marks selections in DIR$
260 LOCAL menu$,scroll_bar$,here$,Line$,width$,height$,margin%
270 REMark LOCAL size_x$,size_y$,char_x$,char_y$,border$,border_colour%
280 REMark LOCAL max_files$,name_limit$,dir_type$,type_mask$,pick_mask%
290 :
300 max_files%="480"*=MaxFiles" :REMark Generous for 720K disks
310 name_limit%=36 :REMark Standard maximum for Qdos directories
320 text_lines%=2 *=TextLines" :REMark Configurable, 1 to 25
330 size_x%="1"*=XcSize" :REMark 0 or 2 in MODE 8/12; 0 to 2 in MODE 4
340 size_y%="0"*=YcSize" :REMark Configurable to 0 or 1 (double height)
350 CSIZE #status$,size_x$,size_y%
360 char_x%=CHAN_W%(#status$,38)
370 char_y%=CHAN_W%(#status$,40)
380 :
390 dir_type%="255"*=DirFileType" : type_mask%=255 : pick_mask%=256
400 DIM dir$(max_files$,name_limit%) :REMark names from 1 onwards
410 DIM flag$(max_files%) :REMark Global for multiple selections
420 :
430 menu%=NEWCHAN$
440 border%="3"*=BorderWidth" : border_colour%="50"*=BorderColour"
450 width%=char_x% * name_limit% + border% * 4
460 height%=char_y% * text_lines% + border% * 2
470 margin%=(40-border%*2)
480 OPEN #menu$,"SCR_" & width% & "x" & height% & "a" & margin% & "x3"
490 :
500 scroll_bar%=NEWCHAN$
510 OPEN #scroll_bar$,"SCR_8x" & height% & "a" & (2+width%+margin%) & "x3"
520 PAPER #scroll_bar$,2
530 width%=width% - border% * 4
540 :
550 GET_DIR
560 here%=0 : Line%=here%
570 TIDY MENU
580 SHELL SORT
590 FILL_WINDOW here%
600 CURSEN #status%
610 CHOOSE
620 CLOSE #scroll_bar%
630 CLOSE #menu%
640 :
650 IF here%=0
660 RETURN ""
670 ELSE
680 RETURN dir$(here%)
690 END IF
700 END DEFine REQUEST$
710 :
720 DEFine PROCedure GET_DIR
730 LOCAL dir$,length$,name$(name_limit%),head$(64)

```

```

740 dir%=ANYOPEN$(path$,4)
750 IF dir%>0
760 PRINT #status%;"Reading directory"!path$
770 free%=1
780 :
790 REPEAT seek
800 IF EOF(#dir%) THEN EXIT seek
810 head$=INPUT$(#dir%,64)
820 IF head$(1 TO 16)=FILL$(CHR$(0),16) : NEXT seek
830 flag$(free%)=CODE(head$(6)) || mark%
840 length%=STRING$(head$(15 TO 16))
850 name%=head$(17 TO 16+length%)
860 IF flag$(free%)=dir_type% : name%=name$ & " ->"
870 dir$(free%)=name$
880 free%=free%+1
890 IF free%>max_files% THEN EXIT seek
900 END REPEAT seek
910 :
920 CLOSE #dir%
930 END IF
940 flag$(0)=dir_type% || pick_mask%
950 IF "-" INSTR path$ = LEN(path$)
960 dir$(0)="-<- DRIVES"
970 ELSE
980 dir$(0)="-<- PARENT"
990 END IF
1000 END DEFine GET_DIR
1010 :
1020 DEFine PROCedure SHELL_SORT
1030 LOCAL i,j,k,n,d,passes,z%,z$(name_Limit%)
1040 n=free%-1
1050 IF n=2
1060 IF dir$(1)>dir$(2)
1070 z$=dir$(1) : z%=flag$(1)
1080 dir$(1)=dir$(2) : flag$(1)=flag$(2)
1090 dir$(2)=z$ : flag$(2)=z%
1100 END IF
1110 END IF
1120 IF n>2
1130 d=1 : passes=0
1140 REPEAT size
1150 d=3*d
1160 IF d>n : EXIT size
1170 passes=passes+1
1180 END REPEAT size
1190 IF passes>2
1200 PRINT #status%;"Sorting"!n!"names in"!passes!"passes"!
1210 END IF
1220 d=d DIV 2
1230 REPEAT pass
1240 PRINT #0!"."!
1250 d=d DIV 3
1260 FOR i=1 TO d
1270 FOR j=1 TO n-d STEP d
1280 z$=dir$(j+d) : z%=flag$(j+d)
1290 FOR k=j TO 0 STEP -d
1300 IF z$>dir$(k) THEN EXIT k
1310 dir$(k+d)=dir$(k)
1320 flag$(k+d)=flag$(k)
1330 END FOR k
1340 dir$(k+d)=z$
1350 flag$(k+d)=z%
1360 END FOR j
1370 END FOR i

```

use of Toolkit commands, but most of these can be replaced by slower SuperBasic equivalents. The rest are widely available or appear in this month's DIY Toolkit column. I shall discuss alternatives as I explain the listing.

The aim is to illustrate SuperBasic programming with a routine that will be useful in almost any program you write. Taskmaster and QJump's Q-Pointer systems include several requesters, but it is not possible to customise these for your own purposes.

Ergon Development use an elegant collection of requesters

in their programs, but these are supplied in Turbo-compiled tasks so you cannot re-use the code. The SuperBasic in Action file requester can be edited and compiled to work any way you like.

In future issues I plan to build a package like DirWork for Qdos, but the first step is to develop a generic SuperBasic file requester. This month's listing sets up the screen, reads and sorts the directory. Next time I shall complete the requester with routines to move around the list and search the system for directory devices.

All Sorts

The requester includes a neat delayed-exchange **Shell Sort** routine, which sorts two arrays in this example but can be adapted simply for other types of data. The Shell Sort is almost as efficient as the Quicksort, the favorite of many computer scientists; in fact it demands less memory and has fewer special cases, often beating the Quicksort when the data is not in genuinely random order.

The basic Shell Sort is well-known, but this implementation is extra-fast because it makes fewer swaps than the usual implementation. The standard Shell Sort works by stepping through the data in decreasing strides, swapping elements as necessary, until the final pass which examines every element, by which

almost in order and few swaps are needed.

The delayed exchange variant uses temporary variables Z\$ and Z% to hold candidates for exchange, and moves them straight to the right place with fewer swaps (or exchanges, or swops if you will). I picked up this idea from a SAM Basic program by mathematician and Sinclair enthusiast W. Ettrick Thomson, and it works every bit as neatly in SuperBasic.

Sorting is a fascinating topic, and the definitive book on the subject is Donald Knuth's ('Newth') **Sorting and Searching** published by Addison Wesley

20 years ago and now available in paperback. It's utterly brilliant but sometimes heavy going unless you're a pure mathematician like the author. Knuth's series **The Art of Computer Programming** is widely held to be the best set of computer science books yet written, and contributed many ideas like heaps and queues to Qdos. Let us know if you'd like to see reviews of the volumes in these pages.

Program notes

Lines 190 and 200 are directives for Turbo, not needed unless you plan to Turbocharge the requester. Two global variables, PATH\$ and MARK%, set the initial state; PATH\$ is the directory to be read, and could be read from the Toolkit 2 default with the DATAD\$ extension. MARK% determines the initial value of the flag for each name in the directory. Set it to 256 and all files will initially be selected, or zero to let the user select from a clean start.

The requester sets up two arrays; DIR\$ holds the name of every file in the directory, from element 1 to FREE%-1. DIR\$(0) holds a dummy heading. The other array FLAG% holds the type of each file and a single-bit flag to indicate that a file is selected. Bitwise logical operators are used to separate the two fields in each integer, saving the need for two arrays.

It would be more elegant to do this with sets and records in a language like Pascal or Modula 2, but SuperBasic has C-style bitwise operators like && and || which make it easy to test, set and extract bits. FLAG% || 256 sets bit 8, without touching the others, while FLAG% && 256 tests just that bit, and FLAG% && 255 extracts the Qdos file type from the low byte. The file type is useful as it discriminates tasks and directories from normal files.

The values 255 and 256 may change if you want to re-assign bits later, so they are given names TYPE_MASK% and PICK_MASK% in the program. This also helps to make it clear what the code is up to.

On return PATH\$ may be


```

1380 IF d=1 THEN EXIT pass
1390 END REPEAT pass
1400 END IF
1410 CLS #status%
1420 END DEFine SHELL_SORT
1430 :
1440 DEFine PROCedure FILL_WINDOW(from%)
1450 LOCAL n
1460 CLS #menu%
1470 FOR n=from% TO MINIMUM%(from%+text_Lines%,free%)-1
1480 ECHO_NAME n
1490 END FOR n
1500 AT #menu%,Line%,0
1510 END DEFine FILL_WINDOW
1520 :
1530 DEFine PROCedure ECHO_NAME(n%)
1540 IF flag%(n%) && pick_mask%
1550 PAPER #menu%,2 : ELSE PAPER #menu%,0
1560 END IF
1570 IF dir_type%=(flag%(n%) && type_mask%)
1580 INK #menu%,7 : ELSE INK #menu%,4
1590 END IF
1600 PRINT #menu%,dir$(n%) : CLS #menu%,4
1610 IF flag%(n%) && pick_mask% : PAPER #menu%,0
1620 END DEFine ECHO_NAME
1630 :
1640 DEFine PROCedure TIDY_MENU
1650 BORDER #menu%,0
1660 CLS #menu%
1670 CSIZE #menu%,size_x%,size_y%
1680 INK #menu%,4 : PAPER #menu%,0 :REMark Beware interaction!
1690 IF border%>1 : BORDER #menu%,border%-1,border_colour%
1700 IF border%>0 : BORDER #menu%,border%,128
1710 SLIDER
1720 END DEFine TIDY_MENU
1730 :
1740 DEFine PROCedure REDRAW
1750 TIDY_MENU
1760 FILL_WINDOW here%-Line%
1770 BLAT Line%
1780 END DEFine REDRAW
1790 :
1800 DEFine PROCedure SLIDER
1810 LOCAL delta
1820 CLS #scroll_bar%
1830 IF text_Lines%>free%-1
1840 depth%=char_y% * text_Lines%
1850 ELSE
1860 depth%=(text_Lines% ^ 2 * char_y%) DIV free%
1870 END IF
1880 IF free%>1
1890 delta=here%-Line% :REMark Ensure decimal division
1900 offset%=(char_y% * text_Lines% * delta )/free%
1910 ELSE
1920 offset%=0
1930 END IF
1940 REMark The slider must be at least two pixels tall
1950 IF depth%<2
1960 depth%=2
1970 IF border%<2 THEN offset%=offset%-2
1980 IF offset%<0 THEN offset%=0
1990 END IF
2000 BLOCK #scroll_bar%,4,depth%,2,offset%+border%,7
2010 END DEFine SLIDER

```

updated. If the result of REQUEST\$ was null, Escape was pressed; otherwise it returns the name of the file picked on the device PATH\$. If multiple files are selected you can find their names by looking for PICK_MASK% bits set in corresponding FLAG% entries.

Lines 270 and 280 comment out LOCAL declarations for pre-set values used in the requester, as most versions of SuperBasic get into a mess if you use more than nine parameters or LOCALs in a single definition. If you use Minerva, an MG rom or any compiler you can re-introduce the lines, making the requester

more modular and easier to merge with other programs.

The DIY Toolkit NEWCHAN% function has a similar purpose, ensuring that the requester does not clobber active channels. You can replace the calls with constant values like 3 and 4 in lines 430 and 500 respectively, as long as you are sure that the calling program does not use these channels.

Window limits

The requester uses three windows: MENU% for the list of names, STATUS% for reports and input, and SCROLL_BAR%

for the sliding indicator alongside the main list.

The character-size and border must leave room for the scroll bar: eight Mode4 pixels, including borders. This rules out only CSIZE 3,0 and 3,1. Double height characters work fine if you make the height of the window a multiple of 20 rather than 10 pixels.

If you're determined to squeeze extra lines onto the screen, use CHAR_INC or _YSTEP to reduce the step between lines, like the Taskmaster FILES menu and Devpac's lower window. The program uses CHAN_W% from DIY Toolkit Volume C to detect the height selected and adjust scrolling automatically.

Alternatively use PEEK_W and CHBASE from DIY Toolkit Volume Q, or a SuperBasic routine like FN Y_SIZE% in the Turbo Toolkit

demos file, which has the weakness that it is upset by Thor and QJump windowing. If all else fails, explicitly state or 'hard-wire' the values that match your chosen character-size but remember to change all the values when you try a new CSIZE!

A second window of two or more lines accepts input and displays status reports. In this example I use the command line window #0, but any other console window of two or more 36+ character lines will do.

Line 600 uses CURSEN from Toolkit 2 to enable the cursor, allowing input from the status window, but you could replace

this with equivalents like CURSOR_ON from Turbo Toolkit, QTRAP #0,14 from DIY Volume T, or an obscure call like PAN #0,0,115.

The new DIY Toolkit extensions ANYOPEN% and INPUTS are used to access the directory in PROCedure GET_DIR. STRING% converts the integer length word into an integer value, but you can safely take the CODE of the second byte HEADS(16) if you do not have Turbo Toolkit or Mark J Swift's PD Toolkit.

Next Month

The listing is complete until it calls CHOOSE at line 610. By that time it should have displayed the scroll bar and first page of the directory. Next month I shall bring the requester to life, with routines like CHOOSE to make selections and move through the list.

SuperBasic In Action

Hardy ints

Perennial Solutions to perennial problems. If you have any favourites, please send them in.

Reading

Stuart Risby's question in Open Channel, QL World January 1993, I started thinking about an answer.

I am a software engineer (and a former hardware design engineer) who is concerned with the problem of data acquisition. I produce commercial software for VAX/VMS systems, but the problem is still the same. How to get data into the computer? I shall describe a few schemes that I hope you will pass on to Stuart.

One approach is to have all input quantities as pulses, so that a single digital pulse represents a unit of electricity (or a unit of temperature, or a Mars bar!) That's simple. All you have to do then is to count the digital pulses. Incidentally, Maplin Electronics, who publish the Maplin Magazine and a catalogue through newsagents, provide a range of circuit boards

for temperature, humidity, light, etc., that could be adapted.

The following techniques all assume that the inputs are digital:

1) Use a QL rom cartridge. By using a buffer (the 74HC244, for instance) and the P/Rom output enable line, it is possible to read an eight-bit quantity into the QL. This byte could be an eight-bit value (such as eight bits from an Analogue to Digital converter) or an eight-bit count (from a counter chip). This only provides a single input in the range 0-255, unless you start using the PROM address lines to select other counter/buffer chips.

This is the simplest way, but it is very QL-specific (it cannot be used on other systems). Typical circuits are shown in the diagrams. The address of the first buffer is starting address of the rom cartridge, and can be read

from SuperBasic by a simple PEEK instruction. IC pinouts are according to the Maplin catalogue.

2) In most of the commercial systems I have dealt with, data is read in via the serial

communications ports of the computer. In these systems, data is collected directly, or via modem, line driver or PAD. So how to read data into the QL using serial ports? This problem has two solutions:

A) Use a simple UART chip such as the 6402. It converts serial data to parallel, and vice versa. So data can be presented on its parallel inputs to be transmitted serially to the QL. Here again we only have eight bits of data, but now we have a choice of does the UART send out data (count value) continuously, or do we request it? (By writing a character first, for example.) As in the rom cartridge solution above, we can use a character sent FROM the QL (which gets converted into eight receive bits) to select a specific counter.

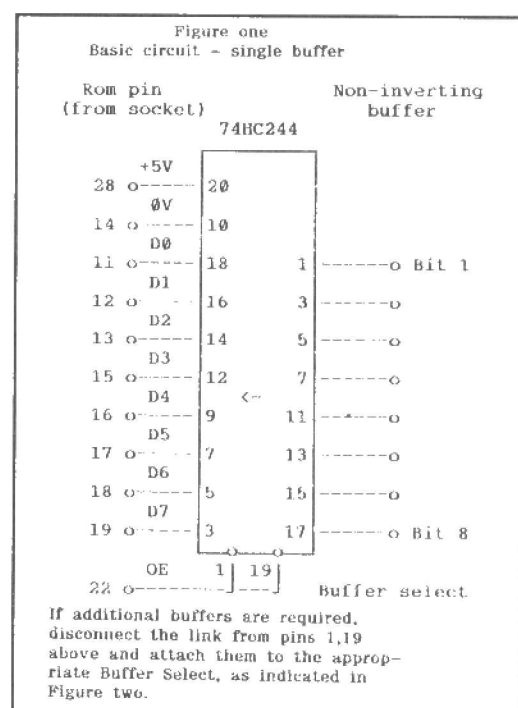
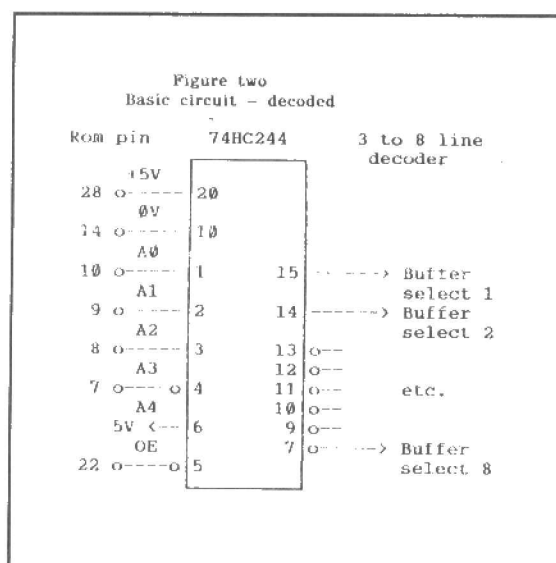
B) The final solution is to use a data acquisition device which collects and records the number of pulses on multiple input channels. The main computer then periodically interrogates the device for data collected over three minutes, half-hour periods, or days.

In the industrial world, these devices are quite expensive, but one of my

pet projects (as yet unfinished) is to build a cheap alternative based on an Intel 8051 micro-controller (enhanced 8048). The 8051 has internal memory, timers, serial communications, and digital ports on-chip. A circuit board of only five or so ICs is enough for the basis of a basic data collector; storing half-hour data for eight channels for ten days. A simple protocol will allow my QL to extract the data using SER1 or SER2 directly in Abacus export format.

All the development so far has been done on my QL using Quill for documents, Conqueror, TASM assembler for MS-DOS, and C69/RATC compilers for software investigations.

Philip Johnson, Stoke on Trent, Staffs



REMINd- Me-Plus 2

Henry Orlowski gets an intelligent reminder from his QL

If you're anything like me you're always getting into trouble because you forget your wedding anniversary or the kids' birthdays or you've missed your dentist's appointment.

The reason you keep missing these important events is because you spend all your spare time on your QL. What you need is some sort of intelligent diary. And if that diary is combined with your QL, and you can call your appointments up on screen any time.

Remind_me_plus2 (Rmp2) is just such an intelligent diary. I call it intelligent because it quickly, even automatically, goes to the right date and gives you your appointments or reminders for the day in question, as well as telling you what you may have missed recently (to see if you can retrieve the situation) and lists other events falling due within the month. Just think! No more missed appointments, an ordered lifestyle and appreciative friends and relatives.

Better diary

But what's the advantage over an ordinary diary? Several. You've probably got a diary which you don't use. This one works on the QL which you use all the time, as part of the suite of applications you run constantly. Regular monthly or annual events such as payment dates or birthdays are entered once only, unlike a diary which you have to change and write up every year. You get a wider perspective of dates in a clear

at-a-glance format. Finally it is very quick to run. No searching for the book and the right page.

The program comes complete with an existing schedule of events which will be of little use to you unless you are or are closely related to author Joe Haftke! However it provides a basic structure and allows you to edit events and dates easily to ones appropriate to you. It also serves as a basic schedule for you to run the program and carry out some demonstrations for familiarisation.

Once you've got the hang of it you're ready to go editing. There is room to store up to 108 individual entries. For the editing process you are presented with 54 at a time on screen. I find it useful at this stage to do a hardcopy, so that you can work out on paper what you wish to put in at the initial set up stage. Once you've got your basic schedule you can just edit the few you need on screen very quickly without having to do a printout every time. The editing option allows you to amend existing entries and add new events. Past events do not need to be taken out unless you run out of spare slots in your schedule.

Autosorting

The schedule allows you to enter the name or description of the event up to 40 characters. The event can be one-off or repeated automatically on a monthly or annual basis. If there are odd events for which you have no

dates you can enter them with an unreal date and the program will report on it every time. This is a useful reminder or messaging system. It doesn't matter which order you enter the events, as the program sorts out the chronology at report runtime.

When you run the program you are first presented with "today's date" to confirm or amend. The only exception is if you are using the Gold Card: if the program detects over 1 megabyte of memory, it assumes you have Gold Card and the battery-backed clock, and that the date must be correct, and it goes straight to the report stage without asking for confirmation. However if you are running Rmp2 as part of a multitasking environment, say with Taskmaster or Qpac 2, you may have less than 1M of memory free even with a Gold Card and the program will ask you to confirm the date. With or without the Gold Card your best bet is to sdate before running.

The program then asks you if you want the report sorted chronologically. The sort is slightly slower but the overall speed makes any differences negligible, especially with the Gold Card. The events are then listed on screen. If you chose to sort, the events you may have missed in the last seven days are listed first in red, followed by forthcoming events in black. A nice touch is that you get the report in the form of x days ago and within x days. Very useful for busy users.

Printout

At this stage you are invited to have a printout of the day's reminders. On my Star LC10

printer I couldn't quite get each item on a single line unless I set condensed mode.

After this you can edit if required. You can print a hardcopy of your schedule in one of two formats. Option 1 is a printout of all 108 events in three columns with the event descriptions truncated to about 10 characters. Blank events are included also. Option 2 allows you to print out the events in a single line per event, untruncated form but does not print out the blank ones. Use the first one when you do your original editing and the second one when you're up and running.

The program can be used in two basic ways. The first is to run it on its own, in which case it will boot up your next application as soon as you have your report and done any editing. This way you've got your reminders or action points at the start of the day. These are the first tasks of the day to work on. Or you can use it multitasking so you can call it up as required during a session.

While the author is publishing and selling Rmp2 himself, I believe the previous versions are still available from Dilwyn Jones Computing. Many people will already have the previous versions, so Joe is making upgrades available (see Information).

Upgrades

There are four major improvements on previous versions and these are worthwhile. The schedule now allows 108 events instead of 54. Event descriptions can now be 40 instead of 30 characters

long. Remind_me_plus also allowed 40 characters, but the report truncated this to 30. Rmp2 gives the full 40 on display. Fourthly and perhaps most importantly the report is now on screen display and hardcopy in the more meaningful chronologically ordered format.

If you are upgrading from Remind_me_plus then your existing events schedule file can be used with the new program. However if you are upgrading from Remind_me you will have to create a modified version. The author includes a utility to carry out the operation, quickly and easily, following the on-screen prompts.

As usual, do not run your program from your master. Make a working copy either with WCOPY or the supplied cloning utility. This only copies the files you need to run the program, and not the additional utilities, so use WCOPY if you want all the files. If you wish to make a configured working copy onto another medium, say mdv instead of flp, then the cloning routine is best. The

routine insists on formatting the new medium, so if you are using preformatted disks or cartridges and want to save a couple of minutes, you can delete lines 155 and 160 from the cloning routine prior to running it.

Error trapping

I liked the ease of use of the program and think it will prove very useful. The manual is concise, and all the options and keypresses are constantly on screen both in the main program and its cloning and

converting routines. Error-trapping is very good. If for some reason it cannot find the events schedule file it will not lock you out, but invite you to place the disk with the file on it into the appropriate drive and try again. To save you as much typing as possible the program presents various previously used defaults at print and edit times. This saves you having to specify baud rate, printer port, month, year etc on every occasion. The program will only suggest saving the events schedule if you have made any changes. When you do a 'save'

you are given the option to have a backup copy. This is a useful provision, and allows you to put a backup disk or tape in the drive for increased security. The save or backup simply overwrites any previous events schedule file that it finds on the device.

One of the author's major objectives was speed and this has certainly been achieved. Loading and reporting are very fast even when running from microdrives, and there is not much of a penalty when requesting an ordered report. Once it's done its job it will run

REMINDERS ON Sun. 04 Apr 1993:-

Test to see if it works.....	6 DAYS AGO	29/3/1993	No.55
MOT EXPIRES.....	3 DAYS AGO	1/4/EACH YEAR	No.2
HOUSEKEEPING DUE.....	3 DAYS AGO	1/0/EACH MONTH	No.18
CAR INSURANCE.....	1 DAY AGO	3/4/EACH YEAR	No.6
PAY CLUB FEES.....	WITHIN 6 DAYS	10/4/EACH YEAR	No.3
SEASON TICKET.....	WITHIN 10 DAYS	14/0/EACH MONTH	No.17
PAY RAC.....	WITHIN 12 DAYS	16/4/EACH YEAR	No.4
TV LICENCE.....	WITHIN 26 DAYS	30/4/EACH YEAR	No.1
PAY QUANTA FEE.....	WITHIN 26 DAYS	30/4/EACH YEAR	No.5

SCHEDULE OF EVENTS

1.TV LICENCE	1000/11/30
2.MOT EXPIRES	1000/14/1
3.PAY CLUB FEES	1000/6/10
4.PAY RAC	1000/7/16
5.PAY QUANTA FEE	1000/12/31
6.CAR INSURANCE	1000/1/1
7.PASSPORT EXPIRES	1998/10/29
8.HOUSE INSURANCE	1000/11/1
9.CONTENT INSURANCE	1000/7/15
10.FREEZER INSURANCE	1000/3/20
11.HEALTH INSURANCE	1000/12/30
17.SEASON TICKET	1000/0/14
18.HOUSEKEEPING DUE	1000/0/1
19.QUANTA AGM	1991/11/24
20.ALL FORMAT SHOW	1991/6/22
21.VISIT FRANK & HELEN	1992/5/6
22.START HOLIDAY	1991/7/8
37.OUR ANNIVERSARY	1000/6/10
38.JOHN'S BIRTHDAY	1000/7/3
39.SUE'S BIRTHDAY	1000/6/5
50.PAY MORTGAGE	1000/6/30
53.Test to see if it works	1993/3/29
54.Stephania's Birthday	1000/12/14

printout
examples

your next program for you without messing around. In fact, sometimes it's over so quickly you want to run it again to make sure you didn't miss anything. No problem. Just leave it in the drive and it will reboot in no time.

I thoroughly recommend Remind_me_plus2 for QL'ers wanting an easy time and date management system on their favourite computer.

REMIND- Me-Plus 2

INFORMATION

Program: Remind_Me_Plus2

Publisher: J Haftke, 7 Lansdowne Road, Sidcup, Kent DA14 4EF. Tel. 081 302 6154.

Price: £20; Upgrade from Remind_Me £10;

Upgrade from Remind_Me_Plus £5.

System: Minerva and Gold Card compatible. Runs on basic QL with microdrives.

VERY BASIC SUPERBASIC

Dilwyn Jones introduces procedures and functions.

This month we'll be learning about procedures and functions. Despite the rather off-putting terms, I hope you will find them useful and easy parts of the QL SuperBasic language.

We studied subroutines in the last issue and found that writing blocks of code as routines which could be called more than once by different parts of the program was useful and convenient. We saved a lot of repetition by doing this, but the snag was that we had to refer to each routine by its line number, which was a bit of a pain, especially in long programs, where we would have to remember where each routine was. After all, humans prefer to think in words and names rather than in computer-style numbers.

Routine names

The QL has a method for giving such routines names. And not just simple, obscure single-character names, but long, meaningful names which describe what the routine does. You can choose quite reasonable names for these routines, as long as they don't clash with

a name which the QL already knows, such as PRINT or LET.

Figure one shows a short example procedure, which simply clears all three windows (or boxes) on the screen with three CLS statements. A procedure is defined by giving it a name and marking the start and end of the routine. It is started simply by using its name elsewhere in the program. It finishes either when it reaches the end of the routine, or when it executes a RETURN statement (which tells it to end early).

The start is marked with a DEFINE PROCEDURE statement followed by the name (in this case CLEAR_ALL_WINDOWS). There must be at least one space between DEFINE and PROCEDURE, and another space between PROCEDURE and the name. The hash symbols after the CLS statements simply refer to which of the boxes on the screen to clear. #0 is the box at the bottom (normally black, where you type the commands), #1 is the red box, while #2 is the part in which programs are listed, coloured white when the QL is in monitor mode, or blue in television mode.

Figure 1

Example procedure to clear all windows on the screen.

```
1000 DEFine PROCedure CLEAR_ALL_WINDOWS
1010   CLS #0 : CLS #1 : CLS #2
1020 END DEFine CLEAR_ALL_WINDOWS
```

Clear names

The end point is marked with the END DEFine statement. Note that in this case it also has the name CLEAR_ALL_WINDOWS after it. This is not essential, the QL will not mind if you just type it as END DEFine or as END DEFine CLEAR_ALL_WINDOWS. It is a little clearer for you the programmer and anyone else who reads it if you do include the name, especially in longer examples.

Note that when typed into the QL, the names DEFine and PROCedure come out in mixed case, that is, some letters are in upper case, while others are in lower case. You have to type those in upper case, but once you have typed those, the QL knows what you mean and inserts the extra lower case letters for you, so you just have to type in DEF PROC, for example, or END DEF and the QL will insert the full names for you. This saves you a little bit of typing.

Having typed in this little routine, all we have to do to clear all the boxes on the screen is to use the command CLEAR_ALL_WINDOWS, like this:

```
100 CLEAR_ALL_WINDOWS
```

In this example, we do not save much by using procedures, but if the routine is called several times in a program it does save having to type in sev-

eral copies of the same commands.

Like a command

Have you noticed that the procedure is called exactly as if it were a new QL command? In a way, it is. By writing programs as procedures, we are in a way extending the computer's vocabulary by teaching it new routines, which we can call by name.

Another good reason for using procedures, quite apart from avoiding repetition, is to split large programs up into smaller, meaningful sections. This simplifies writing large programs, because you can then write and test parts of it at a time.

Figure two shows another example using procedures. This time, we create a routine to roll a dice for use in games. It uses many commands we have already learned about in this series, and it will be developed into examples of more complex use of procedures. The end result of this routine is a word printed on screen - the number one, or two, etc.

Another way in which this routine could be written is as a FUNCTION. Now this complex-sounding term simply means that the routine passes a value back to the calling part of the program, rather than just simply finishing and that's that. The routine works out a value, which it RETURNS (hence the use of the RETurn keyword to pass the

Figure 2 - Dice routine

```
1000 DEFine PROCedure ROLL_DICE
1010 LET random_number = RND(1 TO 6)
1020 SElect ON random_number
1030 =1 : PRINT'One'
1040 =2 : PRINT'Two'
1050 =3 : PRINT'Three'
1060 =4 : PRINT'Four'
1070 =5 : PRINT'Five'
1080 =6 : PRINT'Six'
1090 END SElect
1100 END DEFine ROLL_DICE
```

value back). A couple of examples will explain more than a thousand words in this case. **Figure three** shows the dice routine adapted in such a way that it no longer prints the words, but simply derives a random number to represent the dice and give it back to the part of the program which asked for it. So the routine could be called by different parts of the program and use the number in different

which ends with the dollar symbol, or an integer (whole number only) function by using a name which ends with the percentage symbol. See **Figures four** and **five**. The string function returns the value of the dice as text, One, Two, three, etc. and could be called with a command such as LET a\$ = DICES\$ or PRINT DICES\$.

Note the case of the letters of the word function. You can also

Figure 5 - String function

```
1000 DEFine FuNction DICES$
1010 LET random_number = RND(1 TO 6)
1020 SElect ON random_number
1030 =1 : value$ = 'One'
1040 =2 : value$ = 'Two'
1050 =3 : value$ = 'Three'
1060 =4 : value$ = 'Four'
1070 =5 : value$ = 'Five'
1080 =6 : value$ = 'Six'
1090 END SElect
1100 RETurn value$
1110 END DEFine DICES$
```

and functions is to use mixed case names, where names start with an upper case letter and are in lower case for the rest of the name, eg DEFine PROCedure Roll_Dice. Which method you use is up to you. I always tend to use lower case names for variables and upper case for procedure and function names. Some people prefer to use mixed case names for procedures and functions.

also used by another part of the program and this routine destroys a value set in another part of the program. Oops!

We can specify that a variable used in a procedure or function is "local" to that routine. In other words, it is created temporarily for that routine and it "disappears" after the routine has finished. So in this way, you could have two variables called "random_number" to help prevent the problem just mentioned. So when the function is called, it creates a "temporary" variable called random_number even if one of the same name already exists. When the function has finished, it is deleted again and the old version of the variable is restored. This can be useful in large programs, but in shorter programs it tends to complicate matters more than is usually necessary if there is no use of duplicate names. The LOCAL statement is followed by a list of the names to be treated as local only to that routine, all separated by commas. In theory, you can have as many as you want, in practice, many versions of the QL ROM limit you to about nine local variables. Users lucky enough to have a Minerva system installed on their QL are luckier, the same limits do not apply to them.

Figure 3 - Dice routine written as a function

```
1000 DEFine FuNction DICE_VALUE
1010 LET random_number = RND(1 to 6)
1020 RETurn random_number
1030 END DEFine DICE_VALUE
```

ways. Call this function with a command such as LET r = DICE_VALUE or PRINT DICE_VALUE. The number returned is just like any other QL number and can be stored in a variable or printed, for example.

Types of names

This example was 'floating point' type. You could also write string functions by using a name

enter just the capital letters, or DEF FN if you wish. Many computers use functions, but on most they are abbreviated to FN, the QL uses the full name which is slightly clearer. I have used upper case names for the functions and procedures. There is no real reason to do this, except to avoid mixing the names up with ordinary variables when looking at a program. The QL uses the case in which you first entered that name, so if you entered it in lower case, it is always used in lower case in a program even if you later type it in upper case. Upper case names tend to get mixed up with names of other commands, but normally there is no confusion as long as you know the names of all the commands the computer understands!

Another option for procedures

Figure 4 - Integer function

```
1000 DEFine FuNction DICES%
1010 LET rand_no% = RND(1 TO 6)
1020 RETurn rand_no%
1030 END DEFine DICES%
```

Figure 6 - LOCAL variables

```
1000 DEFine FuNction DICE_VALUE
1010 LOCAL random_number
1020 LET random_number = RND(1 TO 6)
1030 RETurn random_number
1040 END DEFine DICE_VALUE
```


Figure 7 - Parameters

```
1000 DEFine FuNction WHOLE_NUMBER (value)
1010  LOCal added
1020  LET added = value + 0.5
1030  RETurn INT(added)
1040 END DEFine WHOLE_NUMBER
```

"Parameters"

The next trick we can add to these routines is to use "parameters". The bad news is that the terms get even longer and sound more difficult, the good news is that parameters are very useful and powerful facilities.

A parameter is a value which is placed after the name of the procedure or function, in order to pass a value to the routine without having to set variables all over the place. There can be no, one, two or more parameters, depending on what you want to accomplish. As examples of parameters (this is not quite the same, but sufficiently similar for our purposes), consider the PRINT command. You can put variables or strings or numbers after the PRINT command in a list, such as PRINT a,1,'Hello'. You can put such a list after the procedure or function names. In the definition line, the parameters are referred to as 'formal parameters' and the list in the calling routine as 'actual parameters' (I did warn you about complex terms!).

The 'formal parameter' list is a list of variable names in the definition line, usually separated by commas, though other symbols can be used too, all enclosed in brackets. The procedure or function can use look at the values of these in the same way as normal variables, and usually set them as well (though not always). The calling line places the real variables in the corresponding positions and so it is possible, without rewriting a rou-

tine, to call it using different variable names in the list, which is often very useful and versatile.

Brackets or not

When the routine is called, values are inserted for each parameter. When a function is called, all such values must be in brackets, while with a procedure, the actual parameters can be used without brackets. If you put a bracket around one of the procedure parameters when called, it turns that one into an expression, which means that the procedure could not change the value of that variable. This is where life gets complicated, we get into the meanings of terms such as "reference parameters" (which effectively means that the procedure can change the value of that variable when passed) and "pass by value parameters" (the procedure cannot change that value).

Figure seven shows a useful example. This is a function to round off a decimal number to a whole number. SuperBasic already has a built-in function to round a number down to a whole number, INT. This is used in the routine, but can be used on its own like this: LET a = INT(2.5)

Here, we supply INT with an "actual parameter" of 2.5 (it could have been a variable). INT looks at the value passed to it and chops off the digits after the decimal point, then returns the result 2 to be placed in the variable a.

This is all well and good, but sometimes we need to round off a number to the nearest whole

Figure 9 - Multiple parameters

```
100 PRINT TOTAL_OF (2,4,6)
110 ADD 1,2,3 : PRINT g
120 :
1000 DEFine FuNction TOTAL_OF (a,b,c)
1010  RETurn a+b+c
1020 END DEFine TOTAL_OF
1030 :
1040 DEFine PROCEDURE ADD (d,e,f)
1050  LET g = d+e+f
1060 END DEFine ADD
```

number. Simply chopping off the digits after the decimal point is not enough, since a value such as 3.9 would obviously produce the wrong result of 3, whereas it should produce 4. But by adding 0.5 before chopping off the digits after the decimal point, we can ensure that the function returns the result we want, namely rounding off to the nearest whole number, not just down to the nearest whole number. The result is first placed in a variable called "added", which holds the value passed to the routine via the formal parameter "value" with 0.5 added, then INT chops off the digits after decimal point before returning the rounded value to the calling routine. This example can be called with lines such as PRINT WHOLE_NUMBER(3.9) or LET x = WHOLE_NUMBER(y).

Negative numbers

Would that example routine behave as you expected it to with negative numbers? In other words, would you expect it to turn -1.5 into -1 or -2? Try to modify it to produce the result you expect (I'm not saying that one or other is mathematically correct, it is a good exercise to make you think about a tricky question like that).

As a useful demonstration of parameters, that routine can be rewritten as a procedure (see Figure eight). Here, the routine changes the value of a variable parameter instead of returning a function value. Are you confused yet? The routine is now called with an expression such as: LET x = 2.5 : WHOLE_NUMBER x which would change the value

of x to 3. Got it?

But if the expression was written like this:

```
LET x = 2.5 : WHOLE_NUMBER (x)
```

the procedure would be unable to change the value of x since the brackets have turned it into an expression and the value of an expression cannot be set. As an example, think of (x) as x+0. You cannot say in SuperBasic LET x+0 = 2.5.

More than one

To clarify what you must do when more than one parameter is used, Figure nine shows a procedure and function with more than one parameter. Notice where the brackets go around the parameters of the function call, while the procedure call has no brackets around the parameters. Both routines add up three values, the function returns the value in the normal way, while the procedure puts the result in a variable which is later printed by the PRINT command.

I hope you have now learned that while procedures and functions can be difficult to master at first, they are very flexible and useful tools for the program writer and used properly, can save a lot of time both in typing and convenience terms. Properly used, they make programs easier to read and write as long as you take care to break a program down into logically sized chunks. Do take the time and effort to try to master procedures, functions, parameters and local variables, you are missing out on a great SuperBasic facility if you don't!

Figure 8 - Procedures changing parameter values

```
1000 DEFine PROCEDURE WHOLE_NUMBER (value)
1010  LET value = INT(value + 0.5)
1020 END DEFine WHOLE_NUMBER
```

BEGINNER'S MACHINE CODE

In part 5, Alan Bridewell tackles hexadecimal, binary and logic.

In the first part of this series, I mentioned that computers, which are electrical devices, treat high voltages as the number 1 and low voltages as the number 0. By putting these ones and zeros together we can make any number we wish.

We usually work with decimal numbers in everyday life, and computers tend to reflect this. Certainly SuperBasic does. In assemblers we tend to work in hexadecimal numbers. Neither of these is the true number system of a computer. With only two voltages to represent the numbers one and zero, the computer has to work in binary numbers. This is a very easy number system to understand, and it follows the same rules as both the decimal and hexadecimal systems, but with only two numbers, 1 and 0. To represent bigger numbers, a second binary digit will show two, a third will show four, a fourth will show eight, etc.

Bytes and bits

A computer's memory is organised in 'bytes' which consist of eight such binary digits, or 'bits'. The biggest number that can be stored in a byte is when all the bits contain a high voltage to give the number 1: ie 11111111. To see what that number is in decimal we must take into account the weighting of each bit. So, starting from the least significant bit on the right, we get

$$1+2+4+8+16+32+64+128 = 255$$

which shows why the biggest number a byte can store is 255.

By a similar process, you could easily show that a word (two bytes, 16 bits) can store numbers up to 65535, and long words (four bytes, 32 bits) can store numbers up to 4,294,967,295.

When it comes to normal arithmetical processes (add, subtract, multiply and divide), it makes no difference which number system you use for the calculation. You will get the same result. So, for example, the number 15 in decimal is the same as F in hexadecimal, and 1111 in binary. If we double this number, we get:

$$\begin{array}{l} \text{in decimal } 15 + 15 = 30 \\ \text{in hexadecimal } F + F = 1E \\ \text{in binary } 1111 + 1111 = 11110 \end{array}$$

You can convert the numbers yourself if you need convincing that 30 in decimal, 1E in hexadecimal and 11110 in binary are all the same number.

Hexadecimal

SuperBasic programmers use decimal numbers almost exclusively because the language is set up to use it, and it is the most familiar to us. Assembler programmers tend to use hexadecimal because it makes bytes, words and long words easy to deal with, and takes a lot less time and space than binary. You might be wondering at this stage why we need to bother with binary, since we can work in whatever number system suits our needs.

As long as we are treating our data simply as numbers, and are using only add, subtract, multiply or divide, then this is

indeed true. But we often use our data in other ways, where the state of each bit is the important fact, not the overall number they represent. I can illustrate this with a few simple examples.

First, when we send data to the screen ram to produce effects on the screen, it is the state of the individual bits which determines which pixels light up with which colours. Later we shall look at this in some detail. Second, when we send graphics data to a printer, it is the state of the individual bits which determine which pins fire to produce the dots on the paper. Third, when we use a computer to control such things as robots or sound generators, it is the state of the individual bits which determines which circuit is activated to produce the effects required.

Clearly, there are many situations where it is necessary for the programmer to be aware of what is happening to the individual bits, rather than the number they represent. This does not mean that in these situations we can forget about the number they represent, because we can't. The reason for this is when we send the data, or when we receive the data, we do so as a number.

No sense!

To help us deal with bits rather than numbers, there are some important instructions which will make absolutely no sense at all if we treat our data purely as numbers. But they make perfect sense if we treat the data as groups of binary digits. These instructions include

something called the "bit manipulation instructions", the "shift" and "rotate" instructions, and finally the "logical" instructions. We shall deal here with the "logical" instructions, because they are the easiest to understand, and the most likely to be useful to someone just starting out on assembler programming. If, like me, you came to computing via a hobby in electronics, you will recognise these instructions as doing the same thing as the logic chips in the 7400 series TTL chips and the 4000 series Cmos chips.

The first of these instructions is 'NOT', which simply converts each bit into the state it was 'not' in before. If it was zero, it becomes one, and if it was one, it becomes zero. For example, take the line

NOT.W D3

Suppose register D3 contains the word 1100110011001100 in binary. (We could have used NOT.B or NOT.L for either byte or long word sized data if we wished.) Then this line will cause the contents of D3 to change to 0011001100110011. In binary, it's obvious what is happening. But in hex or, even worse, in decimal, there seems to be no connection at all between the number we start with and the one we end up with.

Bit by bit

The next instruction is 'AND'. This instruction compares two numbers bit by bit. If the same bit in both numbers is one, then the result is one, otherwise the result is zero. In other words, we only get one if the bit in the first

number AND the bit in the second number are both one. For example, take the line

```
AND.W D2,D3
```

This will take the word in register D2 and do an AND comparison with the word in register D3, leaving the result of this comparison in D3. Suppose D2 contains 1010101010101010 and D3 contains 1111111100000000 at the start. Then doing the bit by bit comparison, we get

```
D2 at the start
1010101010101010
D3 at the start
1111111100000000
D3 at the end
1010101000000000
```

The third logical instruction is 'OR'. When this compares two numbers bit by bit, the result is one if either of the bits is one, otherwise it gives zero. So, if we take the line

```
OR.W D2,D3
```

and give D2 and D3 the same numbers as before, we get

```
D2 at the start
1010101010101010
D3 at the start
1111111100000000
D3 at the end
1111111110101010
```

The final logical instruction is 'EOR', which stands for 'exclusive OR'. When this compares two numbers bit by bit the result is one if one or the other bit is one, but is zero if both are one, or if neither is one. In other words, it's one if one or the other is one, but not both. So, taking the line

```
EOR.W D2,D3
```

and giving D2 and D3 the same numbers as before, we get

```
D2 at the start
1010101010101010
D3 at the start
1111111100000000
D3 at the end
0101010110101010
```

If all this is rather confusing, you will find that these instructions are available from SuperBasic directly, and it may be instructive to play with them in SuperBasic first to see what they do. If you look in the **QL User Guide** in the **Concepts** section, under the heading **Operators**, you will find the operators '&&' which is the bitwise 'AND', '||' (two sticks) which is the bitwise 'OR', '^' (two hats) which is the bitwise 'XOR' (called 'XOR' in the User Guide), and finally '~' (tilde) which is the bitwise 'NOT'. (Beware: The SuperBasic operators OR, AND, XOR and NOT simply treat the entire number as either zero or non-zero. They do not consider the value of individual bits, and so they are entirely different in their action.)

If you try typing in SuperBasic lines like

```
PRINT - 25 PRINT 120 &&
63 PRINT 25000 || 73 PRINT
1234 ^^ 739
```

...etc

[tilde, two ampersands, two sticks, two hats]

and see what numbers get printed to the screen, you will find out for yourself how these operators work. There are a couple of limitations that must be noted, though. Firstly, the SuperBasic only works with integers, so that the numbers are limited to the range -32768 to 32767. Also, if you use negative numbers, or get negative numbers as a result, you will need to understand how 'signed numbers' work, and this I shall leave until another time.

Demonstration

Continuing with the practice of demonstrating all commands with screen effects, **Listing one** is a demonstration of the effect of these logical operations on the screen ram. The listing actually contains three routines.

The first routine shows the blobs on the screen when numbers are poked into the screen ram, then shows how the application of these logical operators alters the blobs. As indicated in the initial comments, registers D1 and D2 are expected to

contain words for the routine to use. These will be input as parameters in the CALL from SuperBasic. D3 will contain the results of the logical operations. The routine is very simple and repetitive, and if you have followed this series up to now, you will have no problems with it. At various locations down the screen, we display the result of putting into screen ram, in order, the two words, the two words after the 'NOT' instruction, and the results of comparing the two words with the 'AND', the 'OR' and, finally, the 'EOR' instruction. As before, we end up by putting zero in register D0 to make sure we get no error message, and then return to SuperBasic. I will not go through this part in detail, because I think the comments in the listing are explanation enough.

More effects

The first routine demonstrates what the instructions do, but does not give any real indication of the kind of effects that can be achieved. The other two routines are designed to give a taste of this. The second routine copies the screen ram to some other area of ram, called BUFFER, is almost the same as part of the listing in part three of this series.

First we make register D0 into a loop counter by moving \$7FFF, which is one less than the screen ram size, into it. It has to be one less because the DBRA instruction stops looping when the decrementing results in a negative value, not when it results in zero.

(In parts two and three of this series I made the mistake of making the loop counter \$8000 which is equal to the ram size. This meant that the program looped one extra time, which might have caused a crash, but, thankfully, it didn't. I am indebted to Norman Dunbar for pointing this out.)

Next, we make register A0 contain the first address of the screen ram, and register A1 contain the first address of the BUFFER. The loop consists of post-incrementing both the source address and the destination address as we transfer

each byte of the screen ram to the buffer. We end up, as always, by putting zero in register D0 to ensure no error messages occur before returning to SuperBasic.

EOR in effect

The third routine is very similar in that it returns the contents of the buffer back to the screen ram. However, instead of simply copying the bytes it uses the EOR operation to combine the buffer byte with the current screen byte. It does this as follows. First, register D0 is set up as the loop counter, with A1 holding the screen address, and A0 the buffer address. The contents of the address in A0 are moved to register D1, after which A0 is incremented. Then the contents of D1 are EORed with the contents of the address in A1, after which A1 is incremented.

Listing two is a SuperBasic routine to load and run the machine code. Note that the RESPR command allocates enough memory for both the machine code and the buffer, which must be 32K long (the same as the screen ram). After loading the machine code, it asks for two numbers to be input, after which it CALLs the machine code. In order to make sense of what is on the screen, it also PRINTs on the screen next to each blob what produced it. It is set up as an endless loop, so that you can keep trying numbers until you are sick of the sight of it! Pressing Esc will move you on to the next part of the program. Although the CALL command will accept any sized numbers and put sensible parameters into the registers, the SuperBasic bitwise operators will only accept word integers, and, of course, the machine code is written to accept only these. However, the logical operators have long word versions as well if you require them.

After pressing Esc to leave the first part of the program, it will draw a pattern of circles on the screen, and then store this in the buffer. Pressing any key (except Esc) will then clear the

LISTING 1

ALTERING THE DISPLAY WITH BINARY LOGIC

D1 WILL CONTAIN THE FIRST WORD
D2 WILL CONTAIN THE SECOND WORD
D3 WILL CONTAIN THE RESULTS OF LOGICAL OPERATIONS

```

MOVEA.L #82000,A0 ; SCREEN ADDRESS IN A0
MOVE.W D1,A0 ; POKE 1ST WORD TO SCREEN
MOVEA.L #82100,A0 ; NEW SCREEN ADDRESS IN A0
MOVE.W D5,A0 ; POKE 2ND WORD TO SCREEN
MOVE.W D1,D3 ; 1ST WORD TO D3
NOT.W D3 ; NOT 1ST WORD TO D3
MOVEA.L #82200,A0 ; NEW SCREEN ADDRESS IN A0
MOVE.W D3,A0 ; POKE 'NOT' 1ST WORD TO SCREEN
MOVE.W D2,D3 ; 2ND WORD TO D3
NOT.W D3 ; NOT 2ND WORD TO D3
MOVEA.L #822F00,A0 ; NEW SCREEN ADDRESS IN A0
MOVE.W D3,A0 ; POKE 'NOT' 2ND WORD TO SCREEN
MOVE.W D1,D3 ; 1ST WORD IN D3
AND.W D2,D3 ; 1ST WORD 'AND' 2ND WORD IN D3
MOVEA.L #823F00,A0 ; NEW SCREEN ADDRESS IN A0
MOVE.W D3,A0 ; POKE 1ST 'AND' 2ND TO SCREEN
MOVE.W D1,D3 ; 1ST WORD IN D3
OR.W D2,D3 ; 1ST WORD 'OR' 2ND WORD IN D3
MOVEA.L #824000,A0 ; NEW SCREEN ADDRESS IN A0
MOVE.W D3,A0 ; POKE 1ST 'OR' 2ND TO SCREEN
MOVE.W D1,D3 ; 1ST WORD IN D3
EOR.W D2,D3 ; 1ST WORD 'EOR' 2ND WORD IN D3
MOVEA.L #825000,A0 ; NEW SCREEN ADDRESS IN A0
MOVE.W D3,A0 ; POKE 1ST 'EOR' 2ND TO SCREEN
MOVEQ #80,D2 ; NO ERROR MESSAGE
RTS ; RETURN TO SUPERBASIC

```

SAVE THE SCREEN

SAVE SCREEN TO BUFFER

```

MOVE.W #FFFF,D0 ; LOOP COUNTER
MOVEA.L #820000,A0 ; SCREEN ADDRESS IN A0
LEA.A1 BUFFER,A1 ; BUFFER ADDRESS IN A1
; TO (A1), THEN
; INCREMENT BOTH
DBRA D0,LOOP ; IF D0 > 0, THEN DECREMENT D0 AND
; BRANCH TO 'LOOP'
; IF D0 < 0, THEN LEAVE THE LOOP
; AND CONTINUE THE PROGRAM
MOVEQ #80,D0 ; NO ERROR RETURN
RTS

```

FOR SAVE SCREEN WITH CURRENT SCREEN

FOR SCREEN

```

MOVE.W #FFFF,D0 ; LOOP COUNTER
MOVEA.L #820000,A1 ; SCREEN ADDRESS IN A1
LEA.L BUFFER,A0 ; BUFFER ADDRESS IN A0
; TO D1, THEN
; INCREMENT A0
EOR.B D1,(A1)+ ; EOR CONTENTS OF (A1)
; WITH D1, THEN
; INCREMENT A1
DBRA D0,LOOP2 ; IF D0 > 0, THEN DECREMENT D0 AND
; BRANCH TO 'LOOP'
; IF D0 < 0, THEN LEAVE THE LOOP
; AND CONTINUE THE PROGRAM
MOVEQ #80,D0 ; NO ERROR RETURN
RTS

```

LISTING 2

```

100 Z=REFSPR(30280)
110 LBYTES fipl_listing1_code,z
120 OPEN#3,con_612x256a0x2
130 REPEAT loop
140 REPEAT wordsin
150 CLS#3
160 INPUT#3,"1st WORD? (-32768 to 32767) ",word1
170 INPUT#3,"2nd WORD? (-32768 to 32767) ",word2
180 IF word1 > -32768 AND word1 < 32767
190 IF word2 > -32768 AND word2 < 32767 THEN EXIT wordsin
200 END IF
210 END REPEAT wordsin
220 CLS#3
230 CALL z,word1,word2
240 AT#3,3,5:PRINT#3,word1
250 AT#3,3,5:PRINT#3,word2
260 AT#3,6,5:PRINT#3,"NOT ",word1," = ","word1
270 AT#3,9,5:PRINT#3,"NOT ",word2," = ","word2
280 AT#3,12,5:PRINT#3,word1," AND ",word2," = ",word1&word2
290 AT#3,15,5:PRINT#3,word1," OR ",word2," = ",word1|word2
300 AT#3,18,5:PRINT#3,word1," EOR ",word2," = ",word1^word2
310 END IF
320 END IF
330 PAUSE -1
340 IF KEYROW(1)=8 THEN EXIT loop
350 END REPEAT loop
360 CLS#3
370 FOR n = 1E+2 TO 1 STEP 1E-2
380 CIRCLE#3,100*SIN(n),100*COS(n),100*n
390 END FOR n
400 sstore
410 PRINT#3,"This screen has been stored"
420 PRINT#3,"Now press a key to clear the screen"
430 PAUSE -1
440 REPEAT loop2
450 PAPER#3,RND(255)
460 CLS#3
470 STRIP#3,6
480 PRINT#3,"Now press a key to EOR with current screen"
490 PAUSE -1
500 IF KEYROW(1)=8 THEN EXIT loop2
510 srestore
520 PAUSE -1
530 IF KEYROW(1)=8 THEN EXIT loop2
540 END REPEAT loop2
550 CLOSE#3
560 DEFINE PROCEDURE sstore
570 CALL z + 80
580 END DEFINE s
590 DEFINE PROCEDURE srestore
600 CALL z + 124
610 PRINT#3,"Press any key to continue"
620 END DEFINE

```

screen with a random paper colour. A further key press will EOR the stored pattern with the screen colour. This process can be repeated as often as you wish, and Esc will leave the program.

Listing three is the machine code in Marcus and Simon's Hex Loader, for those who do not have an assembler.

Once you have got the hang of these instructions, there are a lot of interesting experiments to do. Here are a few suggestions.

For a start, you could alter Listing two so that the pattern

has circles of different colours, and you could see the effect of EORing different colours with each other. You could next alter the EORB instruction in LOOP2 of **Listing one** to ORB and then AND.B to see the effect of these. They are usually not as interesting as EOR (for reasons which should not be difficult to work out if you have understood this article) and often result in a blank screen. But they sometimes give interesting results.

Quite an interesting effect is produced by writing a routine which will apply the NOT opera-

tion to the entire screen using a DBRA loop.

Black screen!

If you EOR the screen with an identical copy of itself, it makes the screen go black! A second EOR will bring the picture back again. A little thought will show you that this must be so. The only thing you need to know (apart from what EOR means) is that a black screen means the whole screen ram is filled with zeros.

Finally, if you have some saved pictures (especially digitised

video images) which you can LBYTES to the screen, you can produce really startling effects by using these operators to combine two or more pictures. Again, this works best with EOR, but quite useful results can sometimes be obtained with AND and OR.

Of course, these instructions do have other, more serious programming applications, but playing around with the screen display is undoubtedly a very enjoyable way of becoming familiar with them.

Happy coding!

LISTING 3

```
100 REMark Sinclair QL World HEX LOADER v 3
110 REMark by Marcus Jeffery & Simon N Goodwin
120 :
130 CLS: RESTORE :READ space:start=RESPT(space)
140 PRINT "Loading Hex...":HEX_LOAD start
150 INPUT "Save to file?":fs
160 SBYTES fs,start,byte:STOP
170 :
180 DEFINT S%:DEFINT F%:DEFINTAL(x)
190 RETURN CGO(F%*(x)-48-7*(h$(x))"9")
200 END DEFine DECIMAL
210 :
220 DEFine PROCEDURE HEX_LOAD(start)
230 byte=0:checksum=0
240 REPEAT load_hex_digits
250 READ h$
260 IF h$=""EXIT load_hex_digits
270 IF LEN(h$) MOD 2
280 PRINT "Odd number of hex digits in: ",h$:STOP
290 STOP
300 END IF
310 FOR b=1 TO LEN(h$) STEP 2
320 hb=DECIMAL(h$):b=DECIMAL(b+1)
330 IF hb>9 OR hb>15 OR hb<0 OR hb>15
340 PRINT "Illegal hex digit in: ",h$:STOP
350 END IF
360 POKe start+byte,16*hb+lb
370 checksum=checksum+16*lb+hb
380 byte=byte+1
390 END FOR b
400 END REPEAT load_hex_digits
410 READ check
420 IF check<checksum
430 PRINT "Checksum incorrect. Recheck data.":STOP
440 END IF
450 PRINT "Checksum correct. Data entered at: ",start
460 END DEFine HEX_LOAD
470 :
480 REMark Space requirements for the machine code
490 DATA 126
500 :
510 DATA "207C00020200":REMark MOVE.L S#20200,A0
520 DATA "3081":REMark MOVE.W D1,(A0)
530 DATA "207C00021100":REMark MOVE.L #S21100,A0
540 DATA "3082":REMark MOVE.W D2,(A0)
550 DATA "3601":REMark MOVE.W D1,D0
560 DATA "4643":REMark NOT.W D0
570 DATA "207C00020000":REMark MOVE.L #S20000,A0
580 DATA "3683":REMark MOVE.W D3,(A0)
590 DATA "3602":REMark MOVE.W D2,D3
```

```
600 DATA "4643":REMark NOT.W D0
610 DATA "207C00021F00":REMark MOVE.L #S21F00,A0
620 DATA "3083":REMark MOVE.W D3,(A0)
630 DATA "3601":REMark MOVE.W D1,D0
640 DATA "C642":REMark AND.W D2,D0
650 DATA "207C00020000":REMark MOVE.L #S20000,A0
660 DATA "3083":REMark MOVE.W D3,(A0)
670 DATA "B643":REMark MOVE.W D1,D0
680 DATA "207C00020000":REMark MOVE.L #S20000,A0
690 DATA "3083":REMark MOVE.W D3,(A0)
700 DATA "7000":REMark MOVEQ #S00,D0
710 DATA "4E73":REMark RTS
720 DATA "0000":REMark MOVEQ #S00,D0
730 DATA "207C00020000":REMark MOVE.L #S20000,A0
740 DATA "30FA0020":REMark LEA.D (A0),A1
750 DATA "1200":REMark MOVE.B (A0),A1
760 DATA "F100FFFF":REMark LEA.D (A0),A1
770 DATA "7600":REMark MOVEQ #S00,D0
780 DATA "4E73":REMark RTS
790 DATA "208077FF":REMark MOVE.W #S77FF,D0
800 DATA "207C00020000":REMark MOVE.L #S20000,A0
810 DATA "41FA000F":REMark LEA.L (A0),A1
820 DATA "1218":REMark MOVE.B (A0),D1
830 DATA "B018":REMark B0L.B D1,(A1)
840 DATA "5105FFFF":REMark DBRA D0,D0-PE
850 DATA "7600":REMark MOVEQ #S00,D0
860 DATA "4E73":REMark RTS
870 DATA "":REMark
```

DILWYN JONES COMPUTING

41 BRO EMRYS, TAL-Y-BONT, BANGOR,
GWYNEDD, LL57 3YT, GREAT BRITAIN

TELEPHONE: (0248) 354023

QL SOFTWARE

A SELECTION FROM OUR RANGE OF NEARLY 100 PRODUCTS FOR THE QL, NO
ROOM TO ADVERTISE THEM ALL HERE, SO PLEASE ASK FOR OUR
CATALOGUE (PHONE FOR A COPY OR SEND A LETTER WITH YOUR ADDRESS).

EASYPTR III part £40.50

Simplified pointer environment programming. Part I consists of sprite editor, menu editor and superbasic extensions to use in your own programs. Applications created using Easypttr III can be compiled with QLiberator. Requires expanded memory, available on disk only.

EASYPTR III part 2 £20.00

Consists of appendix manager and enhanced toolkit for control of menus etc in your programs. EASYPTR III part 3 £20.00

Consists of Easysource and C library routines etc.

QLIBERATOR £50.00

Superb superbasic compiler, compiles virtually all of basic plus most toolkit commands, etc. Produce faster multitasking code from your basic programs. Compile resident extensions, use overlays etc. with the latest V3.36. Can be mouse controlled. Expanded memory required.

BUDGET QLIBERATOR £25.00

Excellent value, virtually all of superbasic but without some of the additional facilities of the full version. Not mouse controlled. Works on unexpanded QL too.

DJTOOLKIT £10.00

Compact toolkit of BASIC extensions, ideal for use with ALiberator. Really useful programming commands, can be distributed with your compiled programs if you wish. At this price, a bargain! Suitable for unexpanded QL.

LINEDESIGN £100.00

Vector drawing package, uses outline fonts and clipart, move and resize text and graphics without loss of quality. Ideal for making posters, etc. Supplied with huge range of fonts and clipart on TEN disks! The more memory your system has, the better! Available on disk only, can be mouse controlled (including SERmouse).

DATA DESIGN 3 £60.00

Superb, fast pointer driven database with free form field structures, with the option of disk based for large files if required, or smaller files can be kept in memory for speed. You do not have to be able to program this version but if you add the API package, it can be programmed from basic, C, or assembler. Expanded memory required, disk only.

API for Data Design £20.00

QPAC2 £39.95

Tony Tebb's superb pointer environment package. In addition to the pointer environment files themselves, this includes tutorials, extensive manual, files menu, channels and jobs menus, easy switching between jobs, hotkeys, etc. Mouse or keyboard controlled, a good introduction to pointer environment, 256k ram minimum, disk only

QPAC1 £19.95

Ideal companion to QPAC2, consists of small accessory programs such as calculator, calendar, clocks, alarm clocks, typewriter, etc. All can be mouse controlled. Pointer environment files included. Can be used with or without QPAC2. Expanded memory required, disk only.

QTPY2 £29.95

Tony Tebb's spelling checker program. Check spelling as you type OR check existing files retrospectively. User interface allows you to write programs which use the dictionary facilities. English, French and German dictionaries included!

DISA £29.00

Interactive pointer driven machine code disassembler. 256k ram min. Disk only.

MEGAToolKIT £25.00

EPROM VERSION £40.00
Large toolkit with over 200 BASIC extensions, suitable for use with QLiberator or Turbo. Many examples supplied, extensive manual. Suitable for unexpanded QL.

DISCOVER £20.00

The painless way to move files from QL to PC and vice versa. As simple as copying files between two disks. 256k ram min., disk only.

MULTI DISCOVER £30.00

In addition to Discover facilities, also contains CPM, Unix CP10, BBC micro and now Spectrum and SAM Coupe file transfer capability. 256k min. ram, disk only.

TEXTIDY £15.00

Assists Discover with conversion of text files by stripping out control codes etc. 256k ram min.

CONVERT-PCX £10.00

Used with Discover, allows transfer of bit mapped PC clipart graphics in PCX format (a common PC file format) to QL screens or Page Designer pages. 256k ram, disk only.

QL-PC-FILESERVER £24.50

Link a PC and a QL via a serial port cable and use this software to enable the two to communicate - the QL can save its files on a PC's disk systems and print to the PC's serial port using normal basic commands like COPY. Works on unexpanded QL.

BANTER £25.00

Simple to use banner maker which uses outline fonts for good quality large texts. Prints sideways across up to 4 sheets of paper. Simple to use, menu driven, on screen preview before printing, etc. Suits most Epson compatible printers.

IMAGE PROCESSOR 2 £15.00

Easy to use graphics system, featuring usual graphic facilities, pixel zoom editing, image enhancement, mode conversion etc. 512k, disk only.

SCREEN COMPRESSION £10.00

Reduce the amount of storage required by graphics on disk or microdrive - supports several QL formats. 256k, disk only.

SCREEN DAZZLER £15.00

Unlike the usual screen savers, which simply turn off the display when the keyboard is not used for a while, this one can activate various graphical displays to provide an attractive means of preventing screen burn-in, more like the screen savers on other computers. If you have a compiler, you can even write your own savers by following the instructions in the manual. Pointer environment compatible.

SCANNED CLIPART 1 £10.00

NEW! A disk full of compressed scanned pictures (decompression program supplied of course) which can be used in most QL programs (DTP, graphics, etc). Assorted collection, containing many pictures you may not find in other collections. Large number of pictures, a bargain at this price. 128k, disk only.

PRINTERMASTER £20.00

Select printer control codes quickly and simply from a menu to set fonts, page lengths, etc. before printing from programs like Quill, etc. 128k, disk/mdv

SERMOUSE £40.00

Albin Hessler's serial mouse driver system for the QL is now available from DJC complete with a QL style matching black mouse with 9 pin serial connector and UK style port adaptor lead. Version 3 driver software. Can now work with The Painter too. NB POSTAGE CHARGES BELOW

MAGAZINES EX-CGH SERVICES

Ask for a price list of back issues of QL Technical Review, QL Leisure Review and QL Adventurer's Forum (all available at time of writing).

SQUIDQY ROUND THE WORLD £12.50

An arcade game, ideal for the young at heart! 128k

5-GAMES PACK £12.50

5 'thinking' games in one bargain pack, 128k

SUPPLIES

FLOPPY DISKS £0.40

DSHD DISKS £0.70

MICRODRIVES £2.50

DISK LABELS £2.00

On printer roll £2.50

ADDRESS LABELS £2.00

MDV LABELS £2.00

MOUSE MATS £2.50

Disk box dividers £3.00

in stock once more!

TERMS: Discounts - buy 2 programs, claim 5% off each, buy 3 or more, claim 10% off each program. Offer applies to software only. **POSTAGE** - Software is sent post free to UK addresses. Overseas please add £1.00 per program for postage (maximum £3.00). Floppy disks and serial mouse - add postage of £2.50. Labels/mouse mats - add postage of £0.50 per item if only buying these. **PAYMENT** - in UK currency (pounds sterling) only please. Payment by cheque, Eurocheque, Postal Order, cash (send by registered post), or by credit card (Visa/Mastercard/Eurocard/Connect). In case of difficulty contact us first to arrange a payment method if none of these is possible for you. Please make cheques, etc payable to DILWYN JONES COMPUTING (not to any other name or abbreviation please, our bank prefers it that way!).